AD-A093 583 NAVAL POSTGRADUATE SCHOOL MONTEREY CA F/G 5/1
PARAMETRIC SIMULATION OF THE DIRECT SUPPORT MAINTENANCE SYSTEM --ETC(U)
SEP 80 A G LOERCH

UNCLASSIFIED NL

1 of 2
AD
A093583

NAVAL POSTGRADUATE SCHOOL

Monterey, California

THESIS

PARAMETRIC SIMULATION OF THE DIRECT SUPPORT MAINTENANCE SYSTEM IN THE BRIGADE AREA.

by

Andrew George Loerch

September 1980

Thesis Advisor: J. K. Hartman

Approved for public release; distribution unlimited

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER

2. GOVT ACCESSION NO. AD-A093 583

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
PARAMETRIC SIMULATION OF THE DIRECT SUPPORT MAINTENANCE SYSTEM IN THE BRIGADE AREA

5. TYPE OF REPORT & PERIOD COVERED
Master's Thesis; September 1980

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)
Andrew George Loerch

8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Naval Postgraduate School
Monterey, California 93940

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS
Naval Postgraduate School
Monterey, California 93940

12. REPORT DATE
September 1980

13. NUMBER OF PAGES
181

14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)
Naval Postgraduate School
Monterey, California 93940

15. SECURITY CLASS. (of this report)
Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)
Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)
maintenance, simulation, logistics models, combat models

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)
This thesis presents a computer simulation model of the direct support maintenance system in the brigade area. Current and future maintenance doctrine is addressed as background, and is used as a basis for the model. Submodels to generate maintenance workload, perform the maintenance functions, and evaluate attrition of the maintenance units are discussed in detail. A program listing is provided and complete documentation is given. Data

DD FORM 1473, 1 JAN 73 EDITION OF 1 NOV 65 IS OBSOLETE
(Page 1) S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

generated by the model is used as an example of a potential application.

Approved for Public Release: Distribution Unlimited

Parametric Simulation of the Direct Support Maintenance System in the Brigade Area

by

Andrew George Loerch
Captain, United States Army
BS, Polytechnic Institute of Brooklyn, 1974

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL

September 1980

Author: Andrew G Loerch

Approved by: James K. Hartman, Thesis Advisor

S H Parry, Second Reader

G T Howard Acting, Chairman, Department of Operations Research

W M Woods, Dean of Information and Policy Sciences

ABSTRACT

This thesis presents a computer simulation model of the direct support maintenance system in the brigade area. Current and future maintenance doctrine is addressed as background, and is used as a basis for the model. Submodels to generate maintenance workload, perform the maintenance functions, and evaluate attrition of the maintenance units are discussed in detail. A program listing is provided and complete documentation is given. Data generated by the model is used as an example of a potential application.

TABLE OF CONTENTS

I. INTRODUCTION

Future combat on the modern battlefield will be intense and will involve great expenditures of resources. The United States Army may be forced to fight outnumbered and outgunned, and will therefore have to depend on superior training, technology, tactics, and logistics to improve its chance of victory against its enemies.

A major necessity, especially early in the conflict, will be to maintain a high percentage of its combat power in operation against the enemy. As such, inoperable equipment must be returned to battle as quickly as possible. The maintenance units, then, will be a key factor in the outcome of battle.

The purpose of this thesis is to develop a model of the Army maintenance system so that meaningful analysis can be performed concerning its use in combat.

In chapter 2, the maintenance system is defined, both as it operates today and as it will operate in the near future. Two examples of previous modelling efforts are also discussed. In chapter 3, a brief explanation of the maintenance model is given. A detailed explanation of the methodology of the model is given in appendix B. The computer program listing is given in appendix E, and the program is completely documented in appendix C. A sample of

the output of the model is shown in appendix D, and appendix A gives an example of the type of problem the model can help to solve, as well as a set of data generated in a model exercise.

II. PROBLEM DEFINITION

A. INTRODUCTION

The purpose of this chapter is to explain the operation of the United States Army's maintenance system, both how it works now, and how it will work in the future. A clear understanding of this system and the various factors that impact on the system is essential if one is to build a model that will accurately represent the maintenance process.

The maintenance system has been modelled in the past and two examples of these modelling attempts are discussed.

Finally, the requirements for a model that could be used to analyze the operations of maintenance units in combat are presented.

B. THE MAINTENANCE SYSTEM IN THE ARMY TODAY

When the operator of a vehicle observes a malfunction, he reports the problem to the organizational maintenance section in his company. Virtually all company sized units in the Army have organic maintenance capability tailored to the type of equipment the unit has. [1] The mission of these "organizational" or second echelon maintenance sections is to perform scheduled preventitive maintenance and to make minor repairs on the unit's organic equipment when needed. The vehicle would be inspected by these

personnel to locate the source of the problem.

Each component of every piece of equipment in the Army inventory is listed in the technical manual for that piece of equipment. Along with the components list is a Maintenance Allocation Chart, MAC, that specifies the level of maintenance that must be performed on the component. [2] After locating the malfunctioning part, the MAC is checked to see whose job it is to repair the vehicle.

If authorization exists for the job to be done at organizational level, the necessary parts are taken from the relatively small supply of parts the unit has on hand, called its prescribed load list or PLL, or they are requisitioned. The repair would then be made and the vehicle would be returned to service.

If the repair is not authorized to be performed at organizational level, the vehicle would be assigned a priority commensurate with its importance to the accomplishment of the mission of the unit, and would be taken to the next level of maintenance for repair.

In the case of a divisional unit, "direct support" or third echelon maintenance support is provided by the companies of the divisional maintenance battalion. Each brigade of the division is supported by one forward support maintenance company. This company would be located in the brigade support area, known as the brigade trains area, which is doctrinally placed about 25 kilometers to the rear

of the forward edge of the battle area. [3]

The rest of the maintenance battalion including the heavy and light maintenance companies and the missile support company would be located in the Division Support Area, the DSA, in the division rear. With their greater capabilities and their more static situation, they provide backup support for the forward support companies.

The maintenance battalion also provides repair parts (Class IX) support both for its own shops performing repairs, and for those of organizational maintenance activities in supported units of the division. each forward support company has a satellite repair parts supply operation that draws on the main repair parts warehouse in the DSA, and supplies parts to all of its supported units.

When the vehicle arrives at the forward support company, an experienced inspector, usually a staff sergeant performs a complete technical inspection of the vehicle. He then orders the parts necessary for the repair and sends it to a crew of mechanics who will actually do the work. Often repair parts will not be immediately available, or all the crews will be busy. The job would then have to wait.

Waiting jobs are performed in order of priority, both on the basis of prioity that the unit has assigned to the equipment, and on the priority the brigade commander has assigned the units of the brigade.

Once again, as in the case of the organizational units,

the possibility exists that the type of repair needed will not be authorized at the direct support level according to the maintenance allocation chart. In that event, the equipment would be further evacuated to the "general support" maintenance units at corps level by the direct support personnel.

When the item is finally repaired, it must repeat the steps in reverse before it is returned to the user so that all intermediate maintenance levels can complete the work orders that were opened when the job was received.

There is another possible fate that could befall the vehicle as it proceeds through the echelons of the maintenance system. At any level, direct support or higher, an inspector can determine that the piece of equipment is so badly damaged that it would cost more to repair it than it is worth. At this point, he will declare the equipment uneconomically repairable and the owner unit of the equipment would have to requisition a new piece of equipment through the supply system. [4] The item would then become a source of supply parts itself and any serviceable components could then be used to repair other items. This process is called cannibalization or substitution and it is especially useful for repair of items that require infrequently ordered parts that would not normally be stocked in the unit PLL or in the direct support repair parts facilities.

An additional element that must be considered when one discusses maintenance is the Operational Readiness Float program of the division. An Operational Readiness Float, ORF, is an item of equipment that is maintained by the divisional maintenance battalion and is issued to a unit to temporarily replace a like item that needs repair. [4] The Army Regulation governing ORF's specifies a complicated formula for computing the number of float items of each type that the division is authorized to have on hand. The regulation is also quite emphatic about the rules for issuing ORF's in regard to the length of time the needed repair is anticipated to take. Usually, the float items are maintained in the DSA by the heavy and light maintenance companies.

C. PROBLEMS AND PROPOSED CHANGES

Several problems exist in the system today that have arisen during the last two decades as it became more and more evident that U.S. forces would fight outnumbered in the next conflict in Europe.

First, the repair parts supply of the division is not 100 percent mobile. Since it is anticipated that a very short preparation time will be available before the next conflict, the inability of the maintenance battalions to move their warehouse operations will impact heavily on the availability of repair parts.

Second, the fact that our fighting resources are very limited compared to potential adversaries makes the rapid return to battle of repaired equipment an absolute necessity. The delays that are inherent to the present system, especially in the area of transport of unserviceable equipment to maintenance facilities, totally preclude meeting this requirement.

The late classification of uneconomically repairable items is a third problem. Time is wasted on equipment that will not be returned to combat and a source of repair parts that would otherwise be immediately available, is removed.

During the 1973 Mid East War, Israeli maintenance units faced these and other problems and were very successful in dealing with them. In particular they were very adept at making repairs and performing classifications much closer to the forward edge of the battle area than had been thought possible. Direct support maintenance teams and inspectors went to the unserviceable equipment rather than forcing the supported units to bring the items to them. This method of employment greatly improved the performance of routine direct support repairs and cannibalizations could be made quickly to maximize the number of systems available for combat.

As a consequence of Israeli success in this area, many of their techniques have been or will be adopted for use by the U.S. Army, and the organizational changes proposed in

the Division 86 study are reflective of this change of philosophy. [5]

To implement this "fix forward" concept in the U.S. Army, major changes were made to the force structure at the direct support level. The forward support company that was previously only under the operational control of the brigade commander will become organic to the brigade as part of a new Brigade Support Battalion. Included in this battalion as augmentation to the forward support company, will be several new teams called Tank Systems Support Teams and Infantry Systems Support Teams. Each team will provide direct support maintenance as far forward as possible with the former supporting the armor battalions and the latter supporting the mechanized infantry battalions of the brigade task force. Each team will be co-located with the organizational maintenance elements of the battalion to do as much direct support work as possible as close to the battle as possible. Since these teams have very limited repair parts storage capabilities, cannibalization will be a major source of supply of repair parts for the operation of these teams.

D. THE NEED FOR FURTHER ANALYSIS

A major tradeoff has been made by moving maintenance elements forward, that is trading surivability for responsiveness. For a conflict of short duration, fixing

forward might be a better way to proceed. If the conflict continues over a longer period, the increased vulnerability of these forward maintenance elements could result in higher casualties among maintenance personnel and, consequently, a degradation of the maintenance capabilities in the brigade and the division. A need therefore exists for analysis to gain insight into the ramifications of this tradeoff.

While the organizational structure of the direct support maintenance units of the future has been specified completely in the Div 86 report, the tactics that they will use, and the location of these elements on the battle field have not been specifically defined. In fact, the guidance that has been given on this subject is quite nebulous. For example, the Operations field manual, FM 100-5, says the following:

> "...Forward support maintenance companies extend their support to combat units by sending contact teams to work with them. Normally more than half of the repairmen of the company will be out working in the combat area. People, parts, and tools are pushed forward into that forward support area when needed; when no longer needed they are pulled back. Supervised battlefield cannibalization may be used when the parts are not available from the supply system, and an item of equipment can be repaired using parts from other unserviceable equipment..." [6]

Another example of the guidance that has been given concerning the implementation of the fix forward concept

comes from the Operational Concept document for the Division Support Command organization specified in the Div 86 study, prepared by the U.S. Army Logistics Center:

> "...The Forward Maintenance Company establishes a base operation in the brigade trains and sends teams forward to provide close-in support consistent with tactical limitations..." [7]

Perhaps the guidance was made very general so that the flexibility of the brigade commander would not be impaired. Insight into the system, however, would be extremely valuable in assisting commanders in using their maintenance assets in the most efficient and effective manner possible.

E. EXAMPLES OF PREVIOUS MODELLING EFFORTS

The modelling of the combat maintenance process has been very limited in the defense modelling community. Usually the models were used to do one specific study and there is not the proliferation of models at different levels that one finds in the modelling of the combat functions.

An example of such a model is the Balanced Forces or BALFOR model developed by the U.S. Army Concepts Analysis Agency, CAA. [8] This model was developed to examine the impact on the combat service support system of increasing and decreasing the levels of prepositioned war reserves in the European Theatre. BALFOR considers all combat service support functions and does not limit itself to maintenance.

Personnel replacement, medical evacuation, and supply of ammunition and fuel were also considered in the model. BALFOR was made to be compatible with several other CAA models, such as the CEM model, so that outputs from these models could be used as sources of input data to BALFOR. This model is deterministic and extremely fast running, and plays the entire theatre combat service support system.

As a tool for analyzing the new maintenance structure however, BALFOR has some serious shortcomings. It does not play nonavailability of parts, which is a significant factor in determining the time that an item will be inoperable. BALFOR also has the underlying assumption that there will be no attrition or interdiction of the maintenance elements. AS such, it is useless as an analysis tool for examining the new force structure.

A more general model of combat maintenance is the Maintenance Support Concepts, MASC, model developed for the U.S. Army Logistics Center by Braddock, Dunn, and McDonald. [9] This model is the most well known today for analyzing maintenance operations. Originally, it was used to evaluate Operational Readiness Float policies.

MASC is a theatre level model which explicitly plays all theatre maintenance units in detail. It is a stochastic simulation. Input parameters were supplied to the designers by the U.S. Army Ordnance Center and School. These inputs included probability of correct diagnosis and repair, out of

stock probabilities, repair time distributions, waiting parts times distributions, float size and distributions, and rates at which vehicles are rendered uneconomically repairable.

A sensitivity analysis of factors in the simulation showed that the factor that most significantly affected the outcome was the failure rate of the items, followed by out of stock probabilities, washout rate, waiting parts time, and the maintenance float policy. This result is intuitively appealing.

The MASC model seems to represent the maintenance functions very well, but, like the BALFOR model, it fails to portray attrition and interdiction of the maintenance units at all. Also, the proponents of the model admit that the model is very scenario dependent and that only one scenario was played.

In order to evaluate the new force structure and tactics proposed for maintenance elements on the modern battlefield, MASC would have to be extended considerably.

F. REQUIREMENTS FOR MODELLING THE MAINTENANCE SYSTEM IN COMBAT

It is evident that past efforts in modelling maintenance in combat have done a credible job in representing the maintenance functions, but they have been woefully inadequate in portraying the various combat

functions that impact on the performance of the maintenance mission. In the past, this approach might have been satisfactory in that combat service support units were doctrinally located a substantial distance from the fighting. Now, however, with the maintenance units closer to the combat, their mission performance will be significantly affected by combat activities. Consequently, for any model to be a useful tool for performing analyses regarding the maintenance system, it must take into consideration these previously unconsidered factors.

There are three major areas of concern that must be included in modelling maintenance in combat. The first is the input to the system. Both the number of jobs to be performed and the rate they enter the system must be represented as accurately as possible.

Next, the actual maintenance functions must be portrayed. All the inspections, repairs, evacuations, and cannibalizations take time and use up resources. In order to evaluate how the performance of the actual maintenance tasks is affected by other factors, they must be represented in considerable detail.

Finally, the effect of the combat situation on the maintenance units must be considered. When a maintenance unit is attacked, its capability to perform its mission is at least disrupted temporarily and may be degraded permanently. Frequently, moves to alternate positions must

be made to improve responsiveness or to reduce vulnerability. These moves also use time that cannot be spent repairing equipment.

A model that could consider these factors would unquestionably be an asset in doing analyses concerning the maintenance system in the brigade.

III. MODEL DESCRIPTION

A. INTRODUCTION

In this chapter, a model that has been designed as an attempt to meet the requirements of modelling the direct support maintenance system in a combat environment is presented and discussed in general terms. A more detailed model description including a brief explanation of the SIMSCRIPT II.5 programming language is presented in appendix B.

The maintenance model presented here is a stochastic, discrete event simulation implemented in the SIMSCRIPT II.5 programming language. [10] The system modelled is the direct support maintenance system in the brigade area. The structure of the maintenance units was taken from the DIV 86 study Table of Organization and Equipment for the Brigade Support Battalion. The actual distribution of personnel, however, is quite flexible in that the user specifies the types and numbers of repairmen in each maintenance unit.

The model only considers repair of damaged tanks and armored personnel carriers because they have the greatest effect on the outcome of the battle. The damage sustained by these vehicles can be categorized as either firepower damage or mobility damage. The amount of damage in each category is expressed in terms of a proportion. This scheme

for quantifying firepower and mobility damage was chosen in an attempt to be consistant with the damage determination in high resolution combat simulations.

There are two scenario options that can be chosen by the user. The first portrays the trading of space for time by the blue defender. The brigade is divided into two teams that alternately drop back to defend a succession of defensive positions. The second portrays the brigade in a stand and fight posture. These scenarios were chosen as ones that would put the most stress on the maintenance system. The simulation ends when the blue force level is reduced to 25% of its original force level. When this condition is met, the program terminates and the results are output.

B. GENERATING THE WORKLOAD

The first major submodel is concerned with producing damaged vehicles for the maintenance system to repair. This battlefield recovery model, entitled the Parametric Analysis of Recovery and Evacuation of Tracked vehicles model, PARET, was developed by MAJ A.F.Affeldt to analyze battlefield recovery tactics and to determine the heavy equipment transport requirements for a maneuver brigade. [11]

The PARET model plays a series of battles corresponding to the succession of red echelons attacking the defending blue force. In each of these battles, casualties are

assessed and the time necessary to destroy enough red systems to force the red attackers into a defensive position are calculated. A Lanchester type, homogeneous force combat model is used to make these computations.

The proportion of the damaged vehicles for which recovery is to be attempted is computed as the ratio of the time available to perform the recoveries, to the total time necessary to recover all the damaged blue vehicles. As recovery is attempted for each damaged system, attrition of the recovery vehicles is stochastically determined.

As originally written, the PARET model did not consider system failures of combat vehicles due to wear and tear. Since this type of failure will constitute a significant portion of the workload, they were included in this model. The assumption of exponentially distributed failure times with common mean time to failure for all vehicles is made, and system failures are scheduled for each vehicle at the beginning of the simulation. Due to the random nature of drawing the failure times, some of the failures occur before the end of the simulation, and some do not. During the pre-battle period, failed systems are taken directly to maintenance after a short time delay. Once the battle begins, however, the failed systems that require recovery are recovered in the same manner as the combat damaged vehicles.

Since the recovery process is influenced to a great

extent by the onset of darkness, night is represented in the model. Several of the parameter values are altered to take into account the effects that reduced visibility has on the recovery process, and recovery is thereby inhibited.

As each successful recovery is completed, a job is generated for the maintenance system and the attributes describing the job are stochastically determined. These attributes include the owning unit, the workorder number of the job, the firepower and mobility damage percentages, and the vehicle type. The component damage is then computed as a function of the overall firepower and mobility damage values. These component damage percentages are used in the maintenance model to determine if a vehicle is to be a candidate for cannibalization.

Once the job is completely characterized by its assigned attributes, it enters the maintenance system at the forward support detachment that is in support of the battalion that owns the vehicle.

C. PERFORMING THE REPAIRS

The second major submodel represents the actions affecting the vehicle once it enters the maintenance system as a job. Each job is modelled explicitly and its progress is monitored through the system until it is repaired and returned to the fighting force, it is evacuated outside the brigade area to higher levels of maintenance, or it is lost

due to enemy activity affecting the maintenance unit at which it is located.

As the job proceeds through the system, various actions are performed on it. These actions include initial inspection, obtaining repair parts either through the supply system or through cannibalization, transport to higher levels of maintenance, and actual repair itself. The time that each action requires is drawn at random from a beta probability distribution whose parameters are computed from the input data. These input time values are taken from those supplied by the U.S.Army Ordnance Center and School for use in the MASC model. [12]

When the job arrives at a forward support detachment, a determination is made as to whether or not the unit has the capability to repair the type of damage the vehicle has sustained. For instance, if after enemy attack, a maintenance unit no longer has any automotive repairmen, it could not repair a vehicle with mobility damage. If the capability does not exist at the unit to perform the required repair, the vehicle is evacuated to the maintenance company. Otherwise the vehicle undergoes an initial inspection.

Several actions take place during the initial inspection of the vehicle. First, the amount of damage sustained by the vehicle is evaluated and a determination is made as to whether the necessary repairs are authorized at the unit.

Next, the availability of the repair parts necessary to perform the repair is checked. Needed parts that are not on hand in unit supply are either requisitioned or are obtained through cannibalization. Finally, if the repair is authorized and the necessary parts are obtained, a search of the unit is made for a crew of repairmen to actually perform the repair. If either the requirement for parts or a repair crew is not met, the vehicle must wait.

When the requirements are met, work begins. If parts are obtained through cannibalization, the substitution of parts from the source vehicles is performed. After a random repair time, the repair is accomplished and the mobility or firepower damage percentage is reduced to zero, depending on which type of damage the crew is able to repair. If the vehicle has sustained both firepower and mobility damage, two separate repair operations must be performed on the vehicle.

When both the firepower and mobility damage are repaired, the vehicle is returned to the fighting force. The crew that has completed the repair then attempts to find another job to do among those waiting.

There are several instances where an evacuation of the vehicle to a higher level of maintenance is called for. First, jobs at forward detachments that need parts are evacuated to the maintenance company immediately. Second, jobs that have sustained greater damage than the maintenance

unit can fix are sent to higher levels of maintenance. Third, if a maintenance unit is forced to move to an alternate position, vehicles that are not mobile must be evacuated. When a vehicle is evacuated from the maintenance company to the division support area, it is assumed to be lost for the duration of the simulation and it is no longer considered in the model.

D. THE COMBAT ENVIRONMENT

The third and final submodel deals with the actions of the maintenance units, particularly the forward detachments, as the result of enemy activity and the combat situation. The model portrays the movement of the detachments to new positions, as well as attack by the enemy on the detachments.

The distance from the maintenance units to the forward line of troops, FLOT, is monitored throughout the simulation. Any time this distance gets smaller than a user supplied breakpoint value, a move of the unit is triggered. After a delay that corresponds to the time needed for the unit to dismantle the maintenance site, the detachment displaces to a new position at a user supplied speed.

During the move, and during the time it takes the unit to resume its maintenance activities, no jobs are accepted for repair. Already accepted jobs that are mobile, accompany the unit to its new position. Others are either

evacuated to the maintenance company or are destroyed in place.

Upon arrival at the new maintenance site several actions take place before the maintenance mission begins again. Jobs that were supposed to have been supplied with repair parts through cannibalization are checked to be sure the source vehicles have also accompanied the unit on the move. If the source vehicles are no longer present and there are no other source vehicles to supply the appropriate parts, these jobs are evacuated to the rear. Then, the crews at the detachment are matched with jobs to do. This action constitutes a reorganization of effort at the new position. When these actions are accomplished, repair work resumes.

The other aspect of combat that is portrayed in this submodel is the attrition of the maintenance units themselves. The assumption is made in the model that the probability of detection and engagement by the enemy of a maintenance unit is related to how close the unit is to the forward line of troops, and to how many vehicles are present at the unit.

A shaping factor that determines the degree of effect the distance of the unit to the FLOT has on the probability function is input by the user so that a variety of situations can be examined. For example, if the user desires to investigate the case that the only detections that are made by the enemy are visual from the FLOT, a

shaping factor would be selected that would make the probability of detection very high over short distances, but very low over medium or long distances.

This probability is evaluated for each of the forward detachments at the conclusion of each battle sequence. Whether or not the detachments are actually attacked is stochastically determined using this evaluated probability value.

The further assumption is made that the maintenance capability of the unit is proportional to the number of personnel present. Consequently only the personnel casualties and the disruption of the unit as the result of the attack are modelled.

The probability that an individual repairman will become a casualty during an attack is user supplied. Thereby, the user has the ability to determine the severity of the attack expected. Whether or not an individual soldier becomes a casualty is stochastically determined using this input probability value. At the end of the attack, a reorganization takes place, and as many crews as possible are formed from the repairmen left alive. These functioning crews of repairmen are then matched with jobs to be done, and work resumes.

In the event that either or both of the repair capabilities, firepower or mobility, are lost by the unit, jobs requiring the type of repair that the unit can no

longer perform are evacuated to the rear. Also, any job that arrives at the unit needing that type of repair is evacuated immediately.

E. OUTPUT FROM THE MODEL

The output generated from a typical replication of the model is shown in appendix D. This output includes the time values for the various maintenance functions and the corresponding beta distribution parameters under the heading of Input Time Parameters. The input values that deal with the battle and recovery operations, and the maintenance system are also shown. For the most part, the values shown are the ones that were used to develop and test the model.

The attributes of every job that enters the maintenance system are listed under the heading Job List. The attributes include the workorder number, the time the vehicle entered the maintenance system, the vehicle type, the owning unit, whether the vehicle was damaged in combat or was a system failure, the mobility and firepower damage percentages, and the component damage percentages.

The results of each battle sequence are also shown. These include the results of enemy attacks on the forward detachments, the number of recovery vehicles killed, and the number of blue and red systems left.

The job list and battle results are generated during the simulation so that the process can be analyzed in detail. A

set of summary statistics is also generated at the end of each replication, as well as a complete backlog listing of vehicles at the various maintenance units and their job status. It should be remembered that the movement of vehicles through the maintenance system is a very dynamic process, and this particular output gives only an instantaneous view of the system. It is included, however, to give some insight into the distribution of the workload in the system.

Several statistics are also generated that are measures of effectiveness for the system. The results of the recovery operations are shown as the numbers of vehicles recovered and the number of vehicles needing recovery, and the average of these values per battle are also given. In a similar fashion the results of the maintenance operations are displayed. Of these, the most important is average repair cycle time, which is a measure of how long a vehicle remains in the system before it is repaired. Since the purpose of the forward detachments is to shorten this time, this value can be used to compare various employment schemes for the detachments. It should be noted that the repair cycle time is calculated only for the vehicles that are returned to battle.

In an attempt to put the functioning of the maintenance system into the context of its effect on the outcome of the battle, the ratio of red casualties to blue casualties is

computed and displayed. The user should remember, however, that the combat model employed in the maintenance model is very simple. As such, this statistic should not be considered as anything but an extremely gross indicator.

Finally, as a measure of the survivability of the maintenance units, the percentage of repair personnel still alive at the end of the simulation is given. This percentage includes the repairmen in the maintenance company that were not exposed to attack.

IV. CONCLUSIONS AND RECOMMENDATIONS

A. PROGRAM PERFORMANCE

The maintenance model simulation program has shown itself to be extremely fast running and easy to use. It compiles in less than 3 minutes of central processing unit time, and it takes approximately 20 seconds of execution time to complete one replication on the IBM 360-67 at the W.R. Church Computer Center at the Naval Postgraduate School. The program uses approximately 250 kilobytes of core storage. Since only standard data storage procedures with no packing of words were used, the model should be easily transportable to any installation that has SIMSCRIPT II.5 capability.

The program is completely documented in appendix C, and instructions for inputing data are included. The input values shown in the sample output in appendix D are representative of the ones that were used in the design and testing of the program.

Appendix A illustrates the type of application in which the model could be used, and data that was actually generated by the program is presented. The distance of the forward detachments to the forward line of troops was varied over a range of 5 to 30 kilometers, and the preliminary analysis of the data seems to indicate that benefit is

gained by "fixing forward", but the potential for losing maintenance assets to hostile action increases the closer they get to the fighting.

B. RECOMMENDATIONS FOR FUTURE ENHANCEMENTS

As is the case with all models, there are several areas in the maintenance model where improvements in methodology could be made.

One such area is the assignment of the damage attributes to the jobs entering the maintenance system. Presently, damage is determined stochastically using the uniform probability distribution. A refinement of the model could be made by ascertaining the distribution of damage sustained by the combat vehicles. A possible approach to determining the distributions would be to use the Simulation of Tactical Alternative Responses (STAR) model to generate data for this purpose. [13] STAR has the capability to determine the impact locations of shots on combat vehicles and to compute the probabilities of mobility and firepower damage as the result of the impact. These damage probabilities correspond to the damage percentages used in the maintenance model.

Another shortcoming of the maintenance model in its present form is its inability to relate the amount of damage sustained by a vehicle to the time required to repair it.

Finally, the attrition model could be improved by assessing damage to the vehicles in the area of the attack,

including the vehicles organic to the maintenance units. Attrition of the maintenance company should also be considered in the next revision of the model.

The structure of the programming is very flexible, and consequently, the implementation of these model improvements would not be difficult.

APPENDIX A

MODEL EXERCISE

A. INTRODUCTION

In this appendix, the results of a model exercise are shown and analyzed. It should be remembered that these results are just an exercise, and that conclusions cannot really be drawn from them since many of the input variable values that were used to generate the data were educated guesses. The exercise does demonstrate a potential model application.

B. EXPERIMENTAL DESIGN

The analysis technique used is a simple one way analysis of variance. The purpose of the technique is to determine if the mean values of the yield variables that result from different treatments differ from each other in a statistical sense.

Only one of the input variables was changed, that being the initial distance from the forward maintenance detachments to the forward line of troops. Every time a detachment moves to a new location, the distance it moves also corresponds to this value. Four distance values are used: 5, 10, 15, and 30 kilometers. The 30 kilometer distance approximates the situation in which all the

maintenance assets are located at the site of the maintenance company. The others represent utilizing the "fix forward" concept.

The yield values are measures of effectiveness that are computed and displayed by the program. They are the repair cycle time, the percentage of recovered vehicles repaired, the percentage of damaged vehicles repaired, and the percentage of maintenance personnel alive at the end of the simulation.

C. DATA

The following data have been produced from 10 replications of the maintenance model for each of the 4 ranges. The mean value for each treatment is displayed below the columns. The significance level from the analysis of variance, which is the probability of obtaining the data under the null hypothesis that all the means are equal, is also shown for each set of data.

To determine which of the treatment means differed, a studentized range test was performed on each set of data. Mean values marked with asterisks (*) differed from ones not marked in the same manner.

1. Repair Cycle Time Data

Rep	30 km	5 km	10 km	15 km
1	4.49	4.17	4.44	3.97
2	3.78	4.20	3.97	3.79
3	3.96	3.37	3.82	4.16
4	3.71	3.44	4.22	3.91
5	3.94	3.90	3.99	4.04
6	3.98	3.94	3.86	4.22
7	4.64	3.42	4.17	3.71
8	4.62	4.09	4.19	3.75
9	3.79	3.57	3.76	4.47
10	4.02	3.43	4.27	3.86
mean	4.09	3.75*	4.07	3.99

significance level = 0.055

2. Percentage of Recovered Vehicles Repaired

Rep	30 km	5 km	10 km	15 km
1	.383	.340	.604	.531
2	.414	.406	.632	.349
3	.323	.570	.377	.549
4	.413	.399	.554	.462
5	.442	.356	.604	.395
6	.607	.436	.489	.422

7	.603	.230	.543	.392
8	.607	.437	.495	.447
9	.488	.323	.417	.469
10	.504	.446	.447	.337
mean	.478	.394*	.516	.435*

significance level = 0.0238

3. Percentage of Damaged Vehicles Repaired

Rep	30 km	5 km	10 km	15 km
1	.279	.224	.352	.297
2	.216	.276	.245	.258
3	.226	.245	.259	.299
4	.242	.248	.359	.311
5	.226	.208	.264	.341
6	.279	.333	.354	.331
7	.317	.151	.356	.219
8	.279	.330	.204	.265
9	.243	.245	.341	.331
10	.227	.318	.302	.250
mean	.253	.258	.304*	.290

significance level = 0.069

4. Percentage of Personnel Alive

Rep	30 km	5 km	10 km	15 km
1	1.000	.640	.730	.810
2	1.000	.620	.870	.770
3	1.000	.740	.640	.700
4	1.000	.670	.860	.750
5	1.000	.700	.910	.780
6	1.000	.720	.810	.710
7	1.000	.710	.830	.900
8	1.000	.710	.770	.760
9	1.000	.750	.790	.860
10	1.000	.630	.810	.830
mean	1.000*	.689*	.820	.787

significance level approximately 0.0

significance level without considering the 30 km group is also close to 0.0

D. ANALYSIS

The above analyses of variance show that there are significant differences in the performance of the maintenance system as the result of different deployment schemes.

For the statisical tests of the hypotheses that the means of each treatment group were equal, a type I error

rate of 0.1 was chosen because of the relatively small sample size and due to the exploratory nature of the experiment.

In each case the null hypothesis that the means were equal was rejected. Further analysis was performed to determine which of the individual treatments differed. For the repair cycle time data, the value for the 5 kilometer distance was significantly shorter than those for the 30 and 10 kilometer distances. However, the repair cycle time only considers the vehicles that were actually repaired, and the percentage of vehicles repaired that were recovered was significantly higher at the 10 kilometer distance than at the 5 kilometer distance. This shows that the jobs repaired were done faster at the 5 kilometer distance, but more jobs were done at the 10 kilometer distance.

The analysis of the casualty data showed that significantly more maintenance personnel became casualties at the 5 kilometer distance than at the 10 or 15 kilometer distances. This result is intuitively appealing, and it demonstrates the price that has to be paid for the increased responsiveness of the maintenance system.

Overall, the data seems to point to the 10 kilometer distance as being the one that produces the optimal mix of responsiveness and protection for the maintenance assets. This distance would correspond to the forward detachments being located in the vicinity of the organizational

maintenance sections, according to present doctrine.

Once again the reader is reminded that real conclusions cannot be drawn from this data due to the hypothetical nature of the input values used.

APPENDIX B

DETAILED METHODOLOGY OF THE MODEL

A. GENERAL

In this appendix a detailed description of the maintenance model is presented in a form suitable for use by analysts and programmers. Additionally, a brief discussion of the SIMSCRIPT II.5 programming language and its use in the maintenance model is presented.

B. THE USE OF SIMSCRIPT II.5 IN THE MODEL

The SIMSCRIPT II.5 programming language is designed to be used to model discrete event simulations. [10] The language is very readable in that the command structure is more like English than that of many other languages. This feature assists the user in following the flow of the program more easily. The basic elements of the language correspond exactly to those of the basic structure of the event step simulation. They are entities, attributes, sets, and events.

Entities are defined as program elements in the modelled system. In the maintenance model for example, vehicles that need repair, the crews that do the repairs, and the various maintenance units themselves are entities in the system. Each entity of a specific class is differentiated from the

other entities of that class by the values of its attributes. All of the entities in the same entity class have the same attribute names but the values of the attributes might differ. For instance, in the maintenance model, each job entity corresponds to a vehicle. The type of vehicle it is, the amount of damage it has sustained, and the unit that it came from are all attributes of the job. Attributes can have real, integer, or alphanumeric values.

A set is a group of entities with some common property. The maintenance model uses this programming feature in two ways. The first is to denote membership of maintenance crews in the various maintenance units. Second, the jobs that need to be done at a maintenance unit are arranged into sets according to their shop status. For example all the vehicles that are waiting for repair parts belong to one set.

An event in SIMSCRIPT is an occurrence which takes place at a specific time, and results in changing the values of entity attributes, removing or adding entities to sets, creating or destroying entities, and/or scheduling other events to take place at a future time. An example of an event in the maintenance model is the arrival of a job at a maintenance unit. When this event occurs, the job is either inspected immediately, in which case a diagnosis event is scheduled for the job, or the inspection is delayed and the job is added to the waiting inspection set. Events take

place instantaneously and do not consume simulated time.

These data structures greatly simplify the explicit modelling of the progress of each job through the maintenance system. The set, entity, and attribute structure used in the maintenance model is given in appendix C.

C. METHODOLOGY

1. Background

The model of maintenance in the brigade area presented here is a stochastic simulation. Only damage to combat vehicles, tanks and armored personnel carriers, is considered. The type of damage played is divided into two categories, firepower and mobility, and the repairs of these types of damage are done separately.

The tactics used by the supported battalions are specified by the user. There are two options. The first is an effort to trade space for time, where the brigade is divided into two teams that alternately drop back to defend a succession of positions. The maintenance system is heavily taxed by this tactic since there is frequent requirement for the maintenance units to move to alternate positions. The second option is a stand and fight option.

The simulation is ended when the blue force is attrited to 25 percent of its original force level. This stopping rule is written into the program and would require only a minor code modification to change.

Since this model is a stochastic simulation, replication of each experiment several times is desirable and necessary for the purpose of performing statistical analyses on the outputs. To reduce the difficulty of performing replications, a loop is included in the program so that as many replications as desired may be made in the same computer run.

2. Input to the Maintenance System

a. Generating Combat Damaged Vehicles

The first major submodel represents the actual destruction of vehicles in combat and the recovery and evacuation of the damaged vehicles for entry into the maintenance system. This model is the SIMSCRIPT II.5 implementation of the Parametric Analysis of Recovery and Evacuation of Tracked vehicles model, PARET, that was developed by MAJ A.F. Affeldt to investigate battlefield recovery tactics and to determine heavy equipment transport, HET, requirements in the maneuver brigade. [11] The HET requirement routines were not needed for use with the maintenance model and were therefore excluded.

The PARET model plays a series of battles corresponding to the succession of red echelons attacking the blue force. In each of these battles, a BATTLE event is called by the program. Attrition of both blue and red forces is computed on the basis of a homogeneous force, fixed

breakpoint Lanchester type model, under the assumption that all the armored vehicles on the battlefield are "tank killers". This model is described in detail in ref.14. The assumption is made that the ratio of attrition coefficients is equal to the force ratio at the start of each battle. The initial exchange ratio as well as the initial force levels for red and blue are input by the user, and an attrition constant is computed to relate exchange ratios in subsequent battles to the number of blue systems alive. This attrition constant is computed by solving the equation:

X=exp(-(ATT.CONST)(BZERO)) (1)

for ATT.CONST where X is the initial exchange ratio and BZERO is the initial blue force level. Exchange ratios in subsequent battles are computed by substituting the number of blue systems alive for BZERO in equation (1).

The actual battle time is a function of the exchange ratio, the force ratio (red/blue) at the start of the battle, and the breakpoint which is the hypothesized attrition percentage that will force red into a defensive position. The value of this breakpoint is supplied by the user. The battle time is computed as:

```
         C1=X**(-.5)                                   (2a)
C2=ln((-Y(BP)+((1/X)-(Y**2)(1-BP)**2)**.5)/(X**(-.5)-Y)) (2b)
         TB=(C1)(C2)                                   (2)
```

where TB is the battle time, Y is the red to blue force ratio, BP is the breakpoint, and X is the exchange ratio.

After red assumes its hasty defensive position, time elapses until the next echelon closes, reorganization takes place, and the next battle begins. This time for rollup and restart is a function of the echelon spacing, a user input, and the interdicted rate of advance, computed as a product of the user input nominal rate of advance and a stochastically generated interdiction level. The interdiction level is allowed to vary uniformly between 0 and 50 percent. So time for rollup and restart is computed as:

$$TRR=(SPACE.ECH/RI)+TB+U(a,b) \qquad (3)$$

where TRR is the time for rollup and restart, TB is battle time, SPACE.ECH is the distance between red echelons, R is the nominal rate of advance of the next red echelon, and I is the interdiction level. Notice that a uniform random number is drawn to represent the time needed for reorganization before the battle begins again. The limits on this random number, a and b, are 5 and 10 minutes of delay time respectively.

The time available for recovery is then computed as the sum of the battle time and the rollup and restart time less a correction factor which accounts for the time between the start of the battle and the first red casualty. So the recovery time is computed thus:

$$REC.TIME=TB+TRR+C \qquad (4)$$

where REC.TIME is the recovery time, TRR is the time for

rollup and restart, and C is the correction factor.

Since the proportion of damaged vehicles that can be recovered is postulated in the PARET model to be equal to the ratio of the time available to perform recoveries, REC.TIME, to the time required to recover all damaged vehicles, the necessity arises to determine the number of vehicles requiring recovery, and the time needed to accomplish all these recoveries.

Red survivors can be computed as:

R.ALIVE=BP(RZERO) (5)

where R.ALIVE is the number of red survivors, BP is the breakpoint for the red forces, and RZERO is the red force level at the start of the battle.

Using this value, the blue survivors are calculated as:

B.ALIVE=RZERO-(RZERO-R.ALIVE)/X (6)

where B.ALIVE is the number of blue survivors, BZERO is the blue force level at the start of the battle, and X is the exchange ratio. The casualties are easily computed as the difference between BZERO and B.ALIVE.

Not all vehicles are recoverable and some are self or like recoverable. The percentages of unrecoverable and self-like recovered vehicles are user inputs. The number of vehicles needing recovery is computed as:

NR=(1-UNREC-SELF.LIKE)(BZERO-B.ALIVE) (7)

where NR is the number needing recovery, UNREC is the

percentage of unrecoverable vehicles, and SELF.LIKE is the percentage of self or like recovered vehicles.

The time needed to recover all the vehicles needing recovery, TR, is a function of the number needing recovery, NR, the number of recovery vehicles available, NA, the time to hookup at the recovery site, TH, the time to travel to the recovery site, TG, and the time needed to travel from the recovery site to the maintenance collection point, TC. The user must supply both the loaded and unloaded recovery vehicle speeds, CCSL and CCSU respectively, so that TG and TC can be computed. TG and TC are calculated as the ratio of distance to be moved to speed, modified by a disorientation factor, D, which represents the tendency for recovery vehicles to become lost on the battlefield. This factor is a percentage of time added to both travel times and is also supplied by the user. TC, TG, and TR can be calculated as follows:

$$TC=MCPD(1+D)/CCSL \quad (8)$$

$$TG=MCPD(1+D)/CCSU \quad (9)$$

$$TR=(NR/NA)(TG+TH+TC) \quad (10)$$

where MCPD is the distance from the recovery site to the maintenance collection point.

The number of vehicles for which recovery is attempted, RECKS, is then computed as:

$$RECKS=NR(REC.TIME/TR) \quad (11)$$

This procedure is repeated for every simulated

battle.

b. Generating System Failures

The PARET model as originally designed did not consider ordinary breakdowns of equipment due to use. Since these system failures comprise a significant portion of the workload of the maintenance system, they are included in the maintenance model.

Prior to the start of the simulation in the main program, a FAILURE event is scheduled for each piece of blue equipment, independent of the battles. The times of the system failures are assumed to be exponentially distributed, and the mean time to failure for each item is a user input. This mean time to failure is in operating hours, and the proportion of time that the equipment operates is divided into the mean time to failure value to convert it to real time. This proportion is also a user input.

During the time that preceeds the first engagement, the only workload generated is from these system failures. Since the model assumes 100% availability of equipment at the start of the simulation, the further assumption is made that the number of recovery vehicles in the supported units will be more than adequate to handle the evacuation requirements before the actual fighting starts. Therefore the actual recoveries of system failures are not explicitly modelled, and they arrive at the maintenance unit after a short delay of beween two and three hours. Once the

battle sequence begins, however, the system failures that occur are added to the number of casualties assessed in the battle and the recovery of the system failures proceeds in the same way as the recovery of combat damaged vehicles.

The FAILURE event also determines the unit that the vehicle comes from and reduces the number of combatants in that unit. To avoid system failures on vehicles that have already been damaged in combat, the proportion of blue vehicles alive is computed in the FAILURE event and a random comparitor is drawn and compared with the proportion. If the random number is larger than the proportion, it is assumed that the vehicle has already been combat damaged and the system failure is ignored.

c. Making Recoveries

Once the total number of vehicles that need recovery, both system failures and combat damaged vehicles, is known in a specific BATTLE event, a recovery mission is attempted for each. At each attempt, the recovery vehicle and the vehicle to be recovered are vulnerable to attack. The assumption is made that the recovery vehicle will be vulnerable to artillery attack during the trips to and from the battle site, and to direct fire only at the battle site. The probability of kill is postulated to be a function of the times that the recovery vehicle is exposed in each of these phases of its mission. These exposure times are adjusted to take into account the various situational

factors that affect the probability of kill of the recovery vehicle.

During the movement phases of the mission to and from the maintenance collection point, the exposure times are TC and TG as previously computed. The probability of kill is calculated as a function of these times, and the developer of the PARET model postulated the following relation:

$$PK=tangent(time) \qquad (12)$$

This relation gives a monotone increasing function in time, since time is measured in hours and the tangent is computed as if time is measured in degrees. A random comparitor is then drawn to determine if the recovery mission is unsuccessful due to interdiction during the movement phases.

Similarly, the adjusted exposure time on the battle site during the hookup is assumed to be a function of the hookup time, TH, the reciprocal of the red target priority of a recovery vehicle, Z, the probability of incorrect identification of the recovery vehicle, P, and the probability of line of sight, L, which are all supplied by the user, as well as a randomly drawn probability of supression, S. The adjusted exposure time, TE.HOOK, is computed as:

$$TE.HOOK=L(S)(Z)(P)(TH) \qquad (13)$$

This value is then used to calculate the probability of kill using the following hypothesized relation:

$$PK=ABS(1/ln(TE.HOOK)) \qquad (14)$$

Once again a random comparitor is drawn and compared to the value of the probability of kill to determine the success of the mission.

When a recovery mission is determined to be a failure, the model assumes that both the recovery vehicle and the vehicle to be recovered are catastrophically killed and are lost for the rest of the simulation.

d. Determining the Job Attributes

At the conclusion of each successful recovery mission, a BREAK event is scheduled to occur at a uniformly distributed time during the available recovery time. It is in the BREAK event that the various attributes of the recovered vehicle are determined.

After a job entity is created and a workorder number is assigned, a random comparitor is drawn and compared to the proportion of tanks in the force to determine whether the vehicle type is a tank or an armored personnel carrier. The proportion of tanks in the force is user supplied.

The damage sustained by the vehicle is then stochastically determined. The damage to the vehicle is expressed as a number between 0 and 1, and the number is interpreted to be the percentage of major subsystems that have been affected. As such, the range of the possible damage that can be randomly drawn is dependent on the

vehicle type and on how the vehicle was damaged. A system failure, for instance, would probably not be as badly damaged as a vehicle damaged in combat.

There are two of these damage proportion values, corresponding to the mobility functions of the vehicle (MOB.DAM), and to the firepower functions of the vehicle (FP.DAM). These are considered separately in the repair of the vehicle.

In each BREAK event, a damage assessment routine, ASSESS.DAM is called to determine the distribution of the damage in each major subsystem of the vehicle. The subsystem damage values are also expressed in terms of a proportion and these values correspond to the percentage of parts that have been rendered unserviceable in the subsystem. The values are used in the cannibalization routines and are stored in a two dimensional array, DAM.REC, and are indexed by workorder number. The major subsystems for each vehicle with type TANK are:

1 Engine
2 Drive Train (transmission)
3 Cooling System
4 Fuel System
5 Electrical System
6 Track and Suspension
7 Fire Control - Optics
8 Fire Control - Ballistic Computer System

9 Turret

10 Hydraulics

11 Armament

Notice that subsystems 1-6 are mobility related and would therefore be repaired by an automotive crew, and subsystems 7-11 are armament related and would be repaired by an armament crew.

Similarly, the major subsystems for vehicles with type APC are:

1 Engine

2 Drive Train

3 Fuel System

4 Cooling System

5 Electrical System

6 Track and Suspension

7 Fire Control

8 Hydraulics - Cupola

9 Armament

For this vehicle type, the mobility subsystems are 1-6, and the firepower subsystems are 7-9. Notice that the total number of subsystems is different depending on the vehicle type.

A line of output is generated for every job under the heading JOB LIST. The line includes the workorder number, the time the vehicle entered the maintenance system, the owning unit, the vehicle type, the firepower and

mobility damage, and the component damage values.

At the conclusion of the BREAK event, the job is scheduled for arrival at the forward maintenance detachment that is in support of the battalion that owns the vehicle.

d. The Daylight Event

The recovery of combat damaged vehicles on the battlefield is greatly affected by darkness, and as such, event DAYLIGHT is included in the model to represent the effects.

The assumptions are made that the first battle begins at dawn and that there are 15 hours of daylight followed by nine hours of darkness each day. When darkness falls, several of the parameters are changed to reflect the increased difficulty of night recovery operations. These parameters include the cross country speeds of the recovery vehicles, the average hookup time at the recovery site, and the disorientation factor.

Some other parameters that relate to the movement of the maintenance units themselves are also adjusted for the effects of darkness. These are the setup time after a move to a new position, and convoy movement speeds.

The magnitudes of the changes to the parameters in this event routine are written into the program, and changes would require only minor alteration of the coding.

3. Modelling the Maintenance Functions

a. General

The second major submodel portrays the actions affecting the damaged vehicles once they enter the maintenance system as jobs. Each job is a separate and distinct entity, and its progress through the system is modelled explicitly until the job is completed and returned to the combat force, it is evacuated to a higher level of maintenance outside the brigade area, or it is lost due to enemy activity affecting the maintenance units.

As the job proceeds through the system, various maintenance functions are performed on it. These functions include initial inspection, ordering and waiting for the repair parts needed to accomplish repair, repair of the armament and automotive functions of the vehicle, and transport to higher levels of the maintenance system.

The time that each function requires is drawn from a beta distribution, the parameters of which are specified in the input data. For each function, the minimum possible time required to perform the action, the maximum time required, and the average time are entered and stored in the T.ACTION array.

Since the range of the beta distribution is from 0 to 1, these values must be scaled down to fit that range so that the appropriate beta parameters can be obtained.

This procedure for scaling down and fitting the beta distribution is accomplished in the COMP.TIMES routine. The parameters once computed are stored in the A array and are output along with the corresponding T.ACTION vector and the index number of the particular maintenance action. The index numbers and the maintenance functions that correspond to them are:

1 - inspection time at the forward detachments

2 - inspection time at the maintenance company

3 - repair time for automotive jobs

4 - repair time for armament jobs

5 - waiting time for repair parts delivery

6 - movement time from forward detachment to maintenance company

7 - waiting time for evacuation from forward detachment to company

8 - waiting time for evacuation from company to division rear

The actual times that are used in the model are those provided by the U.S. Army Ordnance Center and School. [12] The maintenance times given for repair of the tanks are those postulated for the M60A1 tank and not for the XM1. Similarly the repair times for the armored personnel carrier are based on data for the M113 series and not the new Infantry Fighting Vehicle. The assumption is made however, that the values are close enough to be useful. Since these

values are input as data, it would be very easy to change them if better ones become available.

Each time an event is scheduled and a time is drawn from the appropriate beta distribution, the number drawn is scaled back up to real time scale corresponding to that given in the T.ACTION vector.

When a beta distributed random number is generated, SIMSCRIPT uses its internal gamma random number generator. Occasionally, the parameters for the gamma random number generator are such that, although they are mathematically and theoretically correct as parameters of the gamma distribution, the gamma generator gives an error message stating that the parameters used are incorrect. Consequently the program terminates. For this reason a more robust gamma random number generator routine, GAMMA.F, is included in the program. This routine overrides the internal gamma routine.

b. Arrival and Initial Inspection

At the conclusion of the BREAK event, once the various attributes describing the job are determined, an ARRIVAL event is scheduled immediately for the job, causing it to enter the maintenance system at the forward detachment supporting the battalion that owns the vehicle.

In this ARRIVAL event two determinations are made. First, the remaining capability of the unit to perform the needed repair is evaluated. If, after enemy

attack, no automotive repairmen are available, it would be impossible to repair a vehicle with mobility damage. Consequently the vehicle is evacuated to a higher maintenance level. A MOVE.REAR event is scheduled to accomplish the evacuation and the job is filed in the waiting transport queue, WT.QUEUE.

Second, if the capability to do the type of repair required for the vehicle is present, the availability of an inspector to perform the initial inspection is determined. If he is available, a DIAGNOSIS event is scheduled. Otherwise the vehicle must wait for inspection in the WI.QUEUE.

The actions of the inspector are portrayed in the DIAGNOSIS event. These actions include determination of parts availability either from repair parts supply or by cannibalization, the assignment of a crew of mechanics to perform the repair if the parts are available, and determination of whether or not the repair can be accomplished at that maintenance level.

Parts availability is determined stochastically, and a random comparitor is drawn and gamed against the user input value of the probability that the unit in question has the parts.

If parts are not available through the supply system, a check of the other vehicles present at the maintenance site is made to determine whether parts can be

obtained by cannibalization.

In the event that parts are available from either source, a search of the unit is made to find an idle crew to perform the repair. Since firepower and mobility damage must be repaired by separate crews, the crews are located by their assigned mission. When the appropriate personnel are found, the job is filed either in the ARMAMENT set or the AUTOMOTIVE set and a REPAIR event is scheduled. If no crew with the correct mission type is located, the job is filed in the waiting shop set (WS.QUEUE).

If parts are not available and the vehicle is at a forward detachment, the job is evacuated to the maintenance company and a MOVE.REAR event is scheduled. The assumption is made that the time required for parts to be delivered to the forward detachments would be too long considering the need for the detachments to remain mobile, and considering the short duration of their stay at any one place.

At the maintenance company, if the need arises, parts are ordered and a PARTS.COME event is scheduled. The job is then filed in the WP.QUEUE set corresponding to the group of vehicles that are in waiting parts status.

Finally, The waiting inspection set is checked for any other vehicles that need to be inspected. If there are any there, another DIAGNOSIS event is scheduled and the appropriate job is removed from the waiting inspection set.

c. Obtaining Repair Parts

There are two methods of obtaining the repair parts necessary to perform repairs, through the supply system and through cannibalization.

Parts delivered through the supply system are portrayed in the PARTS.COME event which is scheduled in the DIAGNOSIS event. The assumption is made that a job will not be worked on unless all of the repair parts necessary are present. This assumption precludes repairing mobility damage while parts are still needed to repair firepower damage. As such, when a PARTS.COME event is executed and the parts for a job arrive at the maintenance facility, mobility and firepower damage are again considered separately in locating crews to perform the repairs. The procedure used to find crews is identical to that in event DIAGNOSIS. Repairs are scheduled to occur and the job is filed in the AUTOMOTIVE set or the ARMAMENT set as appropriate. If no crews are available the job is filed in the WS.QUEUE signifying that it is waiting shop.

d. Cannibalization

The other source of repair parts is cannibalization. There are two instances in the process when cannibalization is considered for a vehicle. First, in the DIAGNOSIS event, if the job is determined to need parts that are not present in the unit supply, the other vehicles at the unit for repair are checked as possible sources.

Next, at the conclusion of a REPAIR event, if no other jobs are in a waiting shop staus, a check is made of all the jobs that are waiting for parts or waiting for transport to a higher echelon of maintenance to see if parts can be obtained to perform a repair.

The only vehicles considered as sources for parts are those either waiting for evacuation to the rear or those waiting for parts. All of the other vehicles present at the maintenance site are either waiting for repair or are in the process of being repaired, and no purpose would be served by removing parts from them. The jobs waiting for evacuation cannot be repaired at the unit in question, so nothing is lost by taking the serviceable parts from them. The jobs waiting for parts to arrive cannot be worked on until the parts arrive, so removing parts from them merely increases their wait.

The assumption is made that the inspectors in the maintenance unit would know how each vehicle was damaged and what parts are serviceable and available for cannibalization. Therefore, no time is assessed for the parts availability determination.

When the attributes of each job were first determined in the BREAK event, the damage assessment routine, ASSESS.DAM, was called to stochastically determine the proportion of unserviceable parts in each major subsystem of the vehicle. These proportions are used in the

cannibalization routine, CANNIBAL, to compare parts requirements with availability for each subsystem of the vehicle.

When a job is considered for cannibalization, the CANNIBAL routine checks each subsystem of the job against those in waiting transport status, WT.QUEUE, and then against those in waiting parts status, WP.QUEUE, until the supply of vehicles has been exhausted, or until the parts requirements has been met. This checking procedure entails several steps. First, the particular subsystem of the job is examined to see if parts are needed. Then, if a requirement exists, possible source vehicles are examined to see if the proportion of parts available exceeds the proportion needed. If not, the needed parts are considered not available on that potential source vehicle. If so, a random comparitor is drawn and compared to the difference between the proportion of parts available on the source vehicle, and the proportion of parts needed for the job. This random comparison procedure represents a check to see if the parts needed match the parts available on the source vehicle.

When parts are located, the entity number of the source vehicle is recorded in a two dimensional array, CAN.REC, which is a list of the source vehicles for each subsystem of each job that is to be supplied by cannibalization.

The source vehicles that are waiting transport to the rear have their evacuations cancelled, and they are not rescheduled until the parts are removed in the SUBSTITUTION routine. If the source vehicle is waiting for parts, its PARTS.COME event is cancelled and a new one is scheduled either at the time that the parts to be removed arrive, or at the the time the original parts were to arrive, whichever is later.

Jobs for which parts are located through the cannibalization routine are then placed in a waiting shop status. When a repair is finally scheduled for the job, its non-zero IN.CAN attribute will indicate that a substitution of parts from the source vehicles is required before the repair can be made. The IN.CAN attribute corresponds to a row of the CAN.REC array which contains the list of source vehicles.

e. Performing the Repair

The actual repair of the vehicle is accomplished in the REPAIR event. Each repair has a specified job and a specified crew. The crew will have the capability to repair either firepower damage or mobility damage depending on its assigned mission. Consequently, a vehicle that has sustained both firepower and mobility damage must have two REPAIR events scheduled for it.

If the repair parts for the job are to be obtained through cannibalization, the SUBSTITUTION routine

is called. This routine increases the proportion of unserviceable parts in the appropriate source vehicles by an amount corresponding to the proportion for the job to be repaired. Evacuations are then rescheduled for source vehicles requiring them.

The repair is then performed by setting either the firepower damage attribute, FP.DAM, or the mobility damage attribute, MOB.DAM, to zero depending on the mission of the crew doing the work. If, at the conclusion of the repair, both of these attribute values are zero, the job is considered complete and it is filed in the CLOSED.JOB set. The fighting force is also increased at this time, and the vehicle will participate in the next battle.

The remainder of the REPAIR event entails finding another job for the crew to work on. The first place checked is the group of jobs that are waiting shop. If a job is found that has damage the crew has the capability to repair according to its mission attribute, the job is removed from WS.QUEUE and a REPAIR event is scheduled for it.

If no appropriate jobs are available in the WS.QUEUE, each vehicle in waiting transport status and then each job vehicle in waiting parts status is checked to see if parts can be found through cannibalization, so that work can be done. If any such job can be found, a REPAIR event is scheduled for it. If no jobs can be found that the crew

in question can perform, the OCCUPATION attribute of the crew is changed from busy to idle, and the crew will wait for a job to become available.

f. Evacuations to Higher Levels of Maintenance

There are several reasons why a vehicle would require evacuation to the rear. First, if the damage sustained by the vehicle is greater than the maintenance unit has the capacity to repair, the job must be evacuated. Next, if parts are not available at a forward detachment for a job, it must be evacuated. Finally, if a maintenance unit moves to an alternate position, its jobs must be evacuated.

These evacuations are accomplished in the MOVE.REAR event. If the job is to be evacuated from a forward detachment to the maintenance company, an ARRIVAL event is scheduled to bring the vehicle to the company. If the job is to be evacuated to the division support area, it is filed in the EVAC.JOB set and is no longer considered in the model.

4. Modelling the Combat Functions

a. Movement of the Maintenance Units

The movement of maintenance units to new positions is accomplished in the JUMP and GET.THERE events. During a move, time is expended and no maintenance functions can be performed. Only mobile vehicles accompany the unit on the move. The others are left behind unless they can be

evacuated prior to the unit leaving.

A move is triggered when, in the BATTLE event, a maintenance unit gets too close to the forward line of troops, FLOT. The distance at which the unit is too close is input by the user as the variable B.DIST. The rate of movement of the FLOT during the battle is also a user input as the variable FLOT.MOVE which is expressed in kilometers per hour. Therefore, the distance from the FLOT to the maintenance unit is decreased by that rate multiplied by the time of battle during the BATTLE event.

Additionally, when the option to represent the tactic of trading space for time is selected by the user, a further reduction of four kilometers in the distance from the maintenance unit to the FLOT is made for two of the units as one of the battalion teams moves to an alternate position.

When the maintenance unit moves, the assumption is made that the unit will move to a new position that is the same distance from the old position as the old position was from the FLOT at the start of the first battle.

The speed in which the unit moves to its new position is user input as the variable CON.SPEED. This speed is reduced by event DAYLIGHT during night operations. The time it takes the unit to move is the quotient of the distance moved and the speed, plus the time it takes the unit to setup and resume its maintenance mission. This

SETUP.TIME variable is also a user provided value, and is changed by event DAYLIGHT as well.

The calculation of the movement time is made in the JUMP event, and a GET.THERE event is scheduled using that calculated time. Also, entity records of jobs that do not accompany the unit are removed from the sets to which they belong and are destroyed. During the move, the T.JUMP attribute of the maintenance unit has a nonzero value signifying the time at which the unit will once again begin functioning. No ARRIVAL events will occur for that unit until after that time.

The GET.THERE event marks the resumption of maintenance activities at the new position. Vehicles that have accompanied the unit and are in waiting shop status and need parts from a cannibalization are rechecked to make sure the source vehicles are still present at the unit. If not, and if the parts cannot be obtained from vehicles that are with the unit, the vehicle is evacuated and a MOVE.REAR event is scheduled for it.

Another function performed in the GET.THERE event is the matching of idle crews with jobs in waiting shop status. This action represents the reorganization of effort at the new position. When crews are matched with jobs to be done, repairs are scheduled.

b. Attrition of the Maintenance Units

The detection of the maintenance units by the

enemy, the allocation of fires against them, and the actual attrition of personnel is portrayed in this submodel. The assumption is made that the probability of detection of a maintenance unit by the enemy and the probability that the enemy will decide to engage the unit are functions of how close the unit is to the enemy and of the number of vehicles at the maintenance unit. Therefore the following model is postulated:

$$Pr(engagement)=Pr(engagement/detection)Pr(detection)$$

$$Pr(detection)=exp((-VR)**A)$$

where V is the squareroot of the reciprocal of the number of vehicles present at the unit, R is the distance from the forward line of troops to the maintenance unit in kilometers, and A is a user supplied shaping factor that determines the degree of range dependency of the function. The probability of engagement given the detection of a maintenance unit is assumed to be unity for the model due to the great amount of red artillery available. Several plots of the probability function are presented in figure 1.

At the conclusion of each BATTLE event, routine DET.ALLOC is called, and this probability function is evaluated for each maintenance unit. A random comparitor is then drawn to see if the enemy attacks the unit. If so, routine ATTACK is called.

The assumption is made that the maintenance capability of a maintenance unit is proportional to the

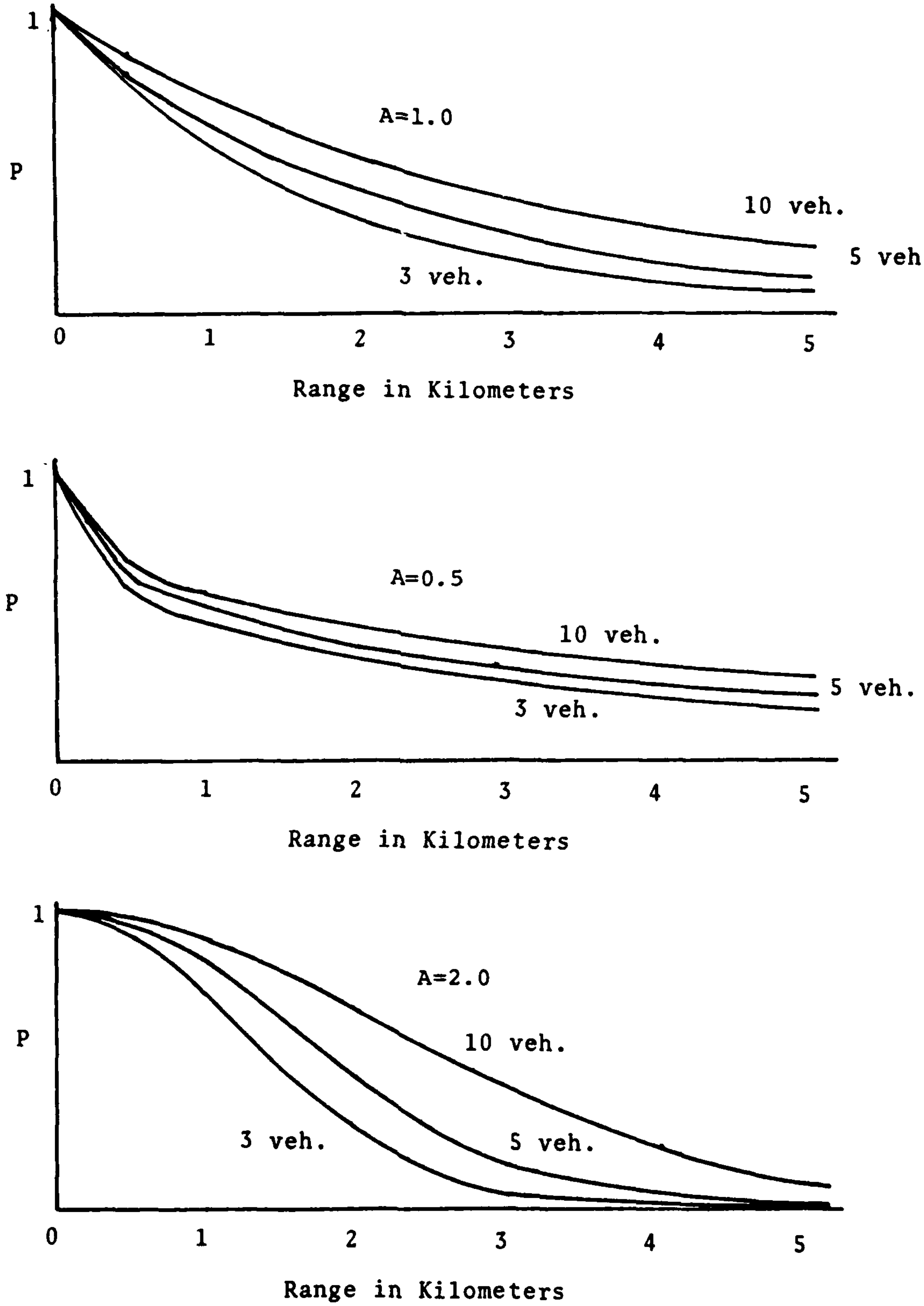

Figure 1. Probability of Detection Plots

number of maintenance personnel present. Consequently only the personnel casualties and the disruption of the maintenance activities of the unit are portrayed.

For each individual person present at the time of the attack, a random comparitor is drawn and compared to the user input value of the probability of kill for personnel, PK.PERS. The program keeps track of the number of each type of repairman present at the unit. These numbers are decreased each time an individual is killed.

It is assumed that in order to function, a crew must have at least two repairmen. Consequently, after the number of kills has been evaluated, a reorganization takes place. In this reorganization as many crews as possible are formed from the repairmen left alive. The rest of the crews have their OCCUPATION attribute value changed to dead, and are no longer considered in the model.

The crews left functioning are then matched with jobs to be done. Repairs in progress by crews that are not killed are delayed until the end of the attack. Similarly, inspections that are taking place at the start of the attack are also delayed.

Finally, in the event that either or both of the repair capabilities, firepower or mobility, are totally eliminated, any job that requires that type of repair must be evacuated to the rear. These jobs are removed from the sets they are filed in, and MOVE.REAR events are scheduled

to effect the evacuations.

If repair capabilities are lost by the unit, any future jobs received will immediately be scheduled for evacuation to the maintenance company. As such, the forward detachment will serve only as a maintenance collection point where damaged vehicles are brought to be sent to the rear.

APPENDIX C

PROGRAM DOCUMENTATION

A. INTRODUCTION

In this appendix, the maintenance model program is fully documented. The set, entity, and attribute structure is described in detail. Each program module is discussed and a line by line explanation of the computer coding is given. Also all variables are defined. The reader is refered to appendix E where a program list is supplied.

B. ENTITY, SET, AND ATTRIBUTE STRUCTURE

1. The MAINT.UNIT Entity

The MAINT.UNIT entity refers to the maintenance units, both the forward detachments and the maintenance company. Each maintenance unit owns the following sets:

SHOP - set of crews in the maintenance unit

WI.QUEUE - set of jobs in waiting inspection status

WP.QUEUE - set of jobs in waiting parts status

WS.QUEUE - set of jobs in waiting shop status

WT.QUEUE - set of jobs waiting transport to the rear

AUTOMOTIVE - set of jobs being worked on by automotive repairmen

ARMAMENT - set of jobs being worked on by armament repairmen

Each MAINT.UNIT entity has the following attributes:

INSPECTOR - number of idle vehicle inspectors

NAME - identification of MAINT.UNIT with the following possible values:

CO.MAINT - defined to mean 0, company

DET1.MAINT - defined to mean 1, detachment 1

DET2.MAINT - defined to mean 2, detachment 2

DET3.MAINT - defined to mean 3, detachment 3

DET4.MAINT - defined to mean 4, detachment 4

VEH.COUNT - number of vehicles present at the unit

D.FLOT - distance from the unit to the FLOT

NM.FOLKS - number of automotive repairmen at the unit

NF.FOLKS - number of armament repairmen at the unit

T.JUMP - the time in which the unit will reach its new location after a move

All the maintenance units belong to a system set called SUP.BN which stands for Support Battalion.

2. <u>The JOB Entity</u>

The JOB entity represents a damaged vehicle that enters the maintenance system to te repaired. Each JOB entity is characterized by the following attributes:

WO.NUM - workorder number of the job

VEH.TYPE - vehicle type; has the following possible values:

TANK - defined to mean 1

APC - defined to mean 2, armored personnel carrier

UNIT - indicates which of the 4 supported battalions the vehicle has come from and will return to when it is repaired

TIME.DOWN - time, measured in days, that the vehicle entered the maintenance system

MOB.DAM - percentage of mobility damage, a number between 0 and 1

FP.DAM - percentage of firepower damage, a number between 0 and 1

T.ARM.REP - time used to repair armament damage

T.AUTO.REP - time used to repair automotive damage

REP.UNIT - NAME attribute of the maintenance unit that repairs vehicle

TOT.DAM - total of FP.DAM and MOB.DAM; not used

IN.CAN - row index of CAN.REC array that lists the source vehicles for cannibalization

CAN.NUM - the number of vehicles to which job provides parts for cannibalization

LOOP.CH - flag to mark vehicle to be removed from set

The JOB entities can become members of various sets as they progress through the maintenance system. These sets include the ones listed under MAINT.UNIT which represent groupings of jobs with the same status. In addition to those sets are three others to which a job may belong. They are:

CLOSED.JOB - the set of jobs for which repairs have been successfully completed

EVAC.JOB - the set of jobs that have been evacuated out of the brigade area

KILL.JOB - used to temporarily hold jobs before they are destroyed at the end of a replication

3. <u>The CREW Entity</u>

The CREW entities represent the groups of repairmen in a maintenance unit. The attributes that describe the crews are:

MISSION - indicates which type of damage the crew can repair; has the following possible values:

AUTO - defined to mean 1, repairs mobility damage

ARM - defined to mean 2, repairs firepower damage

OCCUPATION - indicates what the crew is doing at any time in the simulation; has the possible values:

IDLE - defined to mean 0, waiting for a job to do

BUSY - defined to mean 1, working on a job

DEAD - defined to mean 2, killed in an attack

N.FOLKS - number of repairmen in the crew

Each crew entity belongs to the SHOP set of one of the maintenance units.

C. EVENTS AND ROUTINES

1. The Preamble

The preamble of any SIMSCRIPT II.5 program is used to set up the data structures in terms of entities, sets, and attributes; define event routines and list their arguments; set up the mechanism for collecting statistics by means of the TALLY statements; and define the variables that are global in the program.

The preamble for this program accomplishes these functions and is basically self explanatory.

2. The Main Program

The purpose of the main program is to read data, initialize variables to the appropriate values, create the maintenance unit and crew entities and initialize their attributes, schedule FAILURE events for all the vehicles in the blue force, and schedule the first BATTLE and DAYLIGHT events. The main program also has the replication loop contained in it, which is used to repeat the simulation experiment as many times as the user desires.

Explanation of the coding:

Lines 1-5 reserve core for various arrays

Lines 6-9 define local variables for the main program

Lines 10-11 read input variables

Lines 12-18 generate initial array of random number seeds

Lines 19-32 read input variables

Line 33 calls COMP.TIME routine to read and compute beta parameters

Lines 34-35 initialize variables

Line 36 computes total number of maintenance personnel in system

Line 37 begins replication loop

Lines 38-55 initializes variables for each replication

Lines 56-58 prints initial parameter values on first replication only

Line 59 computes the attrition constant

Lines 60-61 schedules FAILURE events for all blue vehicles

Lines 62-68 initialize variables for replication

Line 69 performs initialization for stand and fight option

Lines 70-76 create the maintenance company and assign attributes

Lines 77-82 calculates number of crews in maintenance company and the number of personnel in each crew

Lines 83-90 creates the crews for maintenance company and assigns attributes

Lines 91-112 repeats maintenance unit and crew creation with attribute assignment for all forward detachments

Line 113 schedules first BATTLE event

Line 114 schedules first DAYLIGHT event

Line 115 prints Job List heading

Line 116 calls Timing Routine and starts the simulation

Line 117 end of replication loop

3. The FAILURE Event

The FAILURE event routine is executed whenever a blue vehicle breaks down due to wear and tear and not as a result of enemy action. The event determines the owning unit of the vehicle and decreases the number of combatants in the unit. To avoid having a system failure for a vehicle that has already been damaged in combat, a random number is drawn and compared to the proportion alive in the unit. If the random number exceeds the proportion alive, the vehicle in question is considered to have already been damaged in combat.

Expanation of the coding:

Lines 2-3 define local variables for the routine

Line 4 checks to see if battle sequence has begun

Lines 5-11 determines unit and schedules BREAK event in pre-battle period

Line 12 determines composition of unit

Line 13 checks to see if vehicle was previously combat damaged

Lines 14-16 decrease unit fighting strength and identifies vehicle as needing recovery

Lines 17-26 repeats 13-16 for split brigade composition

4. <u>The BATTLE Event</u>

The BATTLE event performs several functions in the simulation. It contains the Lanchester formulation that is used to compute the casualties and the time of battle, it performs the recovery missions and determines the attrition of the recovery vehicles, it keeps track of the location of the various maintenance units with respect to the forward line of troops and triggers the units to move if necessary, and it calls the detection and allocation routine which generates enemy attacks on the forward detachments.

Expanation of the coding:

Lines 2-11 define local variables for the routine

Lines 12-13 increment counters

Line 14 increase echelon spacing for divisional spacing

Line 15 prints battle results heading

Line 16 calls routine DET.ALLOC

Lines 17-34 sets parameters depending on the composition of the unit fighting

Line 35 computes exchange ratio for this battle

Lines 36-42 computes the time of battle

Line 43 print WHO.FIGHT and battle time

Line 44 computes interdicted rate of advance of next red echelon

Line 45 computes time for rollup and restart

Lines 46-47 computes time available for recovery

Line 48 computes blue casualties including system failures

Line 49 keeps statistics on blue casualties

Lines 50-51 compute number of vehicles needing recovery

Line 52 checks number of recovery vehicles available

Line 53 computes time needed to recover all vehicles

Lines 54-56 computes the number of vehicles to be recovered

Lines 57-59 calculates exposure times for recovery missions

Line 60 initializes job counter

Line 61 return to regular echelon spacing

Lines 62-86 attempt recoveries for vehicles and determine success or failure of missions

Lines 87-88 print number of recovery vehicles lost

Lines 89-98 schedule BREAK events for system failed vehicles that are still mobile

Lines 99-108 schedule BREAK events for vehicles that are combat damaged but are still mobile

Lines 109-119 schedule BREAK events for vehicles that are system failed and need recovery

Lines 120-127 schedule BREAK events for vehicles that are combat damaged and need recovery

Lines 128-129 update number of blue systems alive

Lines 130-132 update distance of maintenance units to the FLOT

Lines 133-135 collect recovery and casualty statistics

Lines 136-138 update number of red systems in battle

Line 139 return to regular echelon spacing

Line 140 check the composition of the force

Lines 141-152 update distances to FLOT, check to see if distance is too small and schedule a JUMP for the unit if necessary

Lines 153-155 update variables for team 1

Lines 156-168 do the same as lines 141-155 for team 2

Lines 169-175 change distance attributes for all maintenance units

Line 176 print battle results

Line 177 check to see if breakpoint is reached

Line 178 check to see if in split brigade configuration

Lines 179-182 change to combined brigade configuration

Lines 183-184 schedule next battle

Lines 185-187 if in combined brigade configuration and breakpoint has been met, end the simulation and print message

Lines 188-192 if in split bigade configuration and not at breakpoint, change teams and schedule the next battle

Lines 193-194 print Job List heading

5. The BREAK Event

This routine serves the purpose of creating a job entity for each successful recovery mission. Once created, the attributes of the job entity are also determined. These attributes include the workorder number, the vehicle type, and the firepower and mobility damage percentages. The

routine constrains the amount of damage that can be sustained by the vehicle depending on the vehicle type and whether or not the vehicle was damaged in combat. A system failed vehicle will be either mobility or firepower damaged but not both, and a maximum of 0.2 damage percentage is allowed. Also if the vehicle type is armored personnel carrier, its firepower damage percentage can attain a maximum value of 0.2 since the vehicle is still valuable even if it cannot shoot. The component damage is determined by calling the ASSESS.DAM routine. Finally, the job is scheduled for arrival at the maintenance detachment that supports the owning battalion.

Explanation of the coding:

Lines 2-6 define local variables for the routine

Line 7 draw random comparitor

Line 8 increment workorder number counter

Lines 9-14 create a job entity and assign workorder number and assign vehicle type using the random comparitor

Line 15 if the job is a system failure, branch to ON

Lines 16-17 if the job is a combat damaged vehicle, let the maximum possible value of its MOB.DAM attribute be 1.0

Line 18 if the job is combat damaged but still mobile, let the maximum possible value of its MOB.DAM attribute be 0.2

Lines 19-22 randomly draw FP.DAM and MOB.DAM for jobs with vehicle type of TANK and branch to DOWN

Lines 23-25 randomly draw FP.DAM and MOB.DAM for jobs with

vehicle type of APC and branch to DOWN

Line 26 draw random comparitor to determine if a system failed vehicle is either mobility or firepower damaged

Lines 27-29 randomly determine either FP.DAM or MOB.DAM for system failed vehicle

Lines 30-32 call ASSESS.DAM routine to determine component damage

Lines 33-38 print line of output for Job List for TANK jobs

Lines 39-44 print line of output for Job List for APC jobs

Line 45-51 schedule arrival event at the appropriate detachment

6. <u>The ARRIVAL Event</u>

This event represents the arrival of a job at a maintenance unit, either a forward detachment or the maintenance company. If the type of repair needed by the vehicle cannot be performed at the unit, the job is evacuated. Otherwise it is prepared for initial inspection.

Explanation of the coding:

Lines 2-3 definition of local variables for the routine

Line 4 set JOB and MAINT.UNIT entity pointers

Line 5 increment number of vehicles at the unit

Lines 6-7 check to see if unit has personnel to do the type of work necessary

Lines 8-11 schedule a MOVE.REAR to evacuate job if unit is not capable of performing work and put the job in WT.QUEUE

Lines 12 -13 if unit has capability to do repair, check to see if all the inspectors are busy; if so, put the job in WI.QUEUE

Line 14 if an inspector is idle, reduce the number of idle inspectors

Lines 15-23 schedule a DIAGNOSIS event at the appropriate randomly drawn time depending on the unit

7. <u>The REPAIR Event</u>

The REPAIR event is the routine that effects the actual repair of the job. Each REPAIR event has a job and a crew specified in the argument list, and the type of damage repaired is dependent on the MISSION attribute of the crew. As such, vehicles with both mobility and firepower damage require two REPAIR events.

Explanation of the coding:

Lines 2-6 definition of local variables in the routine

Lines 7-9 set JOB and MAINT.UNIT entity pointers

Lines 10-13 check to see if parts are to be provided by cannibalization, if so call SUBSTITUTION routine and exchange parts in source vehicles

Lines 14-15 when crew is automotive, set MOB.DAM attribute to 0.0 and remove the job from the AUTOMOTIVE set

Lines 16-17 when crew is armament, set FP.DAM to 0.0 and remove the job from the ARMAMENT set

Lines 18-28 if job is totally repaired, file it in

CLOSED.JOB set, calculate its DOWN.TIME attribute, increment the number returned to battle, remove job from any other sets it is in, and reduce the number of vehicles at the unit

Lines 29-31 increase number of systems alive in the appropriate team

Lines 32-34 if job is not totally repaired file it in WS.QUEUE

Lines 35-36 if WS.QUEUE is empty, branch to CONTROL

Lines 37-42 search WS.QUEUE for a job the crew can do, if one is found branch to TAKE

Lines 43-45 check to see if WP.QUEUE and WT.QUEUE are empty, if so set OCCUPATION attribute of crew to idle and return

Lines 46-54 check the WP.QUEUE for a job for which parts can be obtained through cannibalization

Lines 55-67 if one is found, call SUBSTITUTE routine to exchange the parts and schedule another REPAIR event

Lines 68-69 if none is found, and job is at the maintenance company, set the OCCUPATION attribute of the crew to idle and return

Lines 70-98 same as lines 37-69 for jobs in WT.QUEUE

Lines 99-115 schedule another REPAIR event for job found in WS.QUEUE

8. <u>The PARTS.COME Event</u>

This event represents the arrival of the parts

needed to repair a particular job. An attempt is then made to find an idle crew to perform repairs. If one is found, a REPAIR event is scheduled. Sometimes, parts are obtained for a job through cannibalization before the parts arrive through the supply system. In this case, the PARTS.COME is ignored.

Explanation of coding:

Lines 2-4 definition of local variables in the routine

Line 5 set JOB and MAINT.UNIT entity pointers

Lines 6-7 check to see if parts have been obtained through cannibalization, if they have return

Lines 8-21 find a crew to repair mobility damage, if one is found schedule a REPAIR

Lines 22-23 if no idle crews available, file the job in WS.QUEUE

Lines 24-38 perform lines 8-23 for firepower damage

9. <u>The DIAGNOSIS Event</u>

This event represents the initial inspection of the vehicle at the maintenance unit. As such, parts availability is checked both through the supply system and through cannibalization, and a crew is located to do the work. If parts and a crew are found, a REPAIR event is scheduled. Otherwise the job is placed in the appropriate set.

Explanation of the coding:

Lines 2-7 definition of local variables in the routine

Lines 8-9 set entity pointers

Lines 10-11 set the number of subsystems (FP) depending on the vehicle type

Line 12 increment number of idle inspectors

Lines 13-15 set the probability of having parts and percentage of damage to be fixed depending on whether the job is at the company or at a detachment

Line 16 check to see if vehicle damage exceeds amount of damage that can be fixed, if so branch to EVAC.MAYBE

Lines 17-18 draw a random comparitor and compare to probability of having parts

Lines 19-22 if parts not available call CANNIBAL routine to try to find them by cannibalization, if not available branch to EVAC.MAYBE

Lines 23-36 having parts, find crew to repair mobility damage and schedule a REPAIR event; if none found file in WS.QUEUE

Lines 37-50 having parts, find crew to repair firepower damage and schedule a REPAIR event; if none found file in WS.QUEUE

Lines 51-58 if no parts are found and job is at a detachment, schedule a MOVE.REAR event to evacuate, file in WT.QUEUE, and branch to NEXT

Lines 59-63 if vehicle is too badly damaged, evacuate it and schedule a MOVE.REAR event, file in WT.QUEUE

Lines 64-68 if no parts available at company, schedule a PARTS.COME and file in WP.QUEUE

Lines 69-82 schedule another DIAGNOSIS event for idle inspector if possible

10. The MOVE.REAR Event

The purpose of the MOVE.REAR event is to evacuate jobs to higher echelons of maintenance. Sometimes, if a job becomes involved in a cannibalization, it will have been removed from the WT.QUEUE. In this case, the MOVE.REAR is ignored. If a vehicle is evacuated from the maintenance company, it is filed in the EVAC.REAR set and is no longer considered in the simulation.

Explanation of the coding:

Lines 2-3 definition of local variables in the routine

Line 4 set entity pointers

Line 5 if job is not in the WT.QUEUE, ignore the MOVE.REAR

Line 6 reduce number of vehicles at unit

Line 8-15 schedule an ARRIVAL at the maintenance company for jobs at a detachment

Lines 16-25 file job in EVAC.JOB and remove it from all other sets

11. The DAYLIGHT Event

The DAYLIGHT event represents the reduced capabilities of recovery vehicleles, and the reduced convoy

speeds of moving units during periods of reduced visibility. The routine changes the values of several of the variables back and forth between their day and night values.

Explanation of the coding:

Line 2 return if battle sequence has not begun

Lines 3-12 change from day values to night values

Lines 13-14 schedule daybreak in 9 hours and return

Lines 15-23 change from night values to day values

Lines 24-25 schedule nightfall in 15 hours and return

12. <u>The JUMP Event</u>

This event portrays the movement of a forward detachment to a new position, and the suspension of maintenance activities during the move. Also, vehicles that are not mobile do not accompany the unit on the move, and are either evacuated or destroyed. The time for the unit to move and setup the new maintenance site is calculated and a GET.THERE event is scheduled for the end of that time period.

Explanation of the coding:

Lines 2-3 definition of local variables in the routine

Line 4 set MAINT.UNIT entity pointer

Line 5 compute time it takes the unit to move

Line 6 schedule a GET.THERE event

Lines 7-13 remove and destroy non-mobile jobs in WS.QUEUE

Lines 14-19 remove and destroy non-mobile jobs in WI.QUEUE

AD-A093 583 NAVAL POSTGRADUATE SCHOOL MONTEREY CA F/G 5/1
PARAMETRIC SIMULATION OF THE DIRECT SUPPORT MAINTENANCE SYSTEM --ETC(U)
SEP 80 A G LOERCH

UNCLASSIFIED NL

2 of 2
AD
A093583

END
DATE
FILMED
2-81
DTIC

Lines 20-32 remove and destroy all jobs in WT.QUEUE that will not be evacuated by the time the unit moves

Lines 33-47 remove and destroy all non-mobile jobs in ARMAMENT set and reschedule REPAIR events for jobs that are mobile for a time after arrival at the new site

Lines 48-57 remove and destroy jobs in AUTOMOTIVE set

Line 58 set T.JUMP attribute of maintenance unit to delay job arrivals at the unit

13. <u>The GET.THERE Event</u>

This event marks the resumption of maintenance activities by the maintenance unit at the conclusion of a move. A check is made to make sure that the source vehicles have accompanied the unit on the move, for any job involved in a cannibalization. The crews are then matched with jobs to be done and the maintenance mission is resumed.

Explanation of the coding:

Line 2 definition of local variables in the routine

Line 3 set MAINT.UNIT entity pointer

Line 4 set T.JUMP attribute to 0.0

Lines 5-11 check to see if cannibalizations are still possible

Lines 12-23 schedule a MOVE.REAR event for each job that can no longer be cannibalized

Lines 24-37 match jobs with automotive crews and schedule REPAIR events

Lines 38-51 match jobs with armament crews and schedule REPAIR events

14. The STOP.SIMULATION Event

As its name implies, the purpose of this event is to end the simulation and output the results. The routine is called at the end of each replication and prints out summary statistics, the backlog of jobs at every maintenance unit, and lists of all the jobs that were successfully completed and evacuated out of the brigade area. The START.OVER routine is also called to reset the variable values for the next replication.

Explanation of the coding:

Lines 2-3 definition of local variables in the routine

Lines 4-6 compute the number of repairmen alive at the end of the simulation

Lines 7-10 compute summary statistics

Lines 11-19 print summary statistics

Lines 20-72 print the backlog for each maintenance unit

Lines 73-78 print list of successfully completed jobs

Lines 79-83 print list of jobs evacuated outside the brigade area

Line 84 call START.OVER routine

15. The START.OVER Routine

This routine is called from the STOP.SIMULATION

event to initialize the program for a new replication. All counters are set back to zero, the totals of statistics kept by the program are reset, all scheduled events are cancelled, all jobs in the system are filed in the KILL.JOB set and then destroyed, the CAN.REC and DAM.REC arrays are set to zero, and the maintenance units are destroyed.

Explanation of the coding:

Lines 2-6 zero variables

Lines 7-9 reset system statistic keeping routines

Lines 10-32 cancel and destroy all scheduled events

Lines 33-78 remove all jobs from the sets they are in and file them in the KILL.JOB set

Lines 79-80 destroy the jobs in the KILL.JOB set

Line 81 zero the CAN.REC array

Line 82 zero the DAM.REC array

Lines 83-85 destroy the MAINT.UNIT entities

16. <u>The ASSESS.DAM Routine</u>

The ASSESS.DAM routine is called by the BREAK event to stochastically determine the component damage of a job as a function of the previously computed FP.DAM and MOB.DAM values. The FP.DAM and MOB.DAM values correspond to the proportion of major subsystems of each type that have been damaged. This routine randomly picks the subsystems that are damaged. A random number from between 0.0 to 1.0 is drawn and is assigned to the chosen subsystem. This number

represents the proportion of damaged parts in the subsystem. The values are stored in the DAM.REC array and are indexed by the workorder number of the job. The component damage values are used in the CANNIBAL routine.

Explanation of the coding:

Lines 2-3 definition of local variables in the routine

Line 4 set JOB entity pointer

Lines 5-6 set number of subsystems depending on vehicle type

Line 7 set local variable equal to workorder number of job

Lines 8-9 compute number of damaged mobility subsystems if any

Line 10 draw a number corresponding to one of the subsystems

Lines 11-12 if the particular subsystem has not been chosen before, draw a random number and assign the damage percentage

Line 13 increment counter and repeat sequence until values are obtained for all damaged subsystems

Lines 14-21 repeat sequence for firepower subsystems

17. <u>The CANNIBAL Routine</u>

This routine is called in order to determine if parts can be obtained for a job through cannibalization. This is done by comparing the component damage values of the job with those of potential source vehicles. If a source of

parts is found, the entity number of the source job is recorded in the CAN.REC array. The routine returns to the calling event with a number corresponding to the number of source vehicles if enough were found, or zero if the cannibalization attempt was unsuccessful.

Explanation of the coding:

Lines 2-9 definition of local variables in the routine

Line 10 set JOB and MAINT.UNIT entity pointers

Line 11 set workorder number of job in temporary location

Lines 12-13 set number of subsystems depending on vehicle type of job

Line 14 reserve core for temporary storage of entity numbers for source vehicles

Lines 15-17 count number of damaged subsystems of job

Lines 18-19 check each job in WT.QUEUE as a possible source vehicle

Line 20 loop through all subsystems

Line 21 branch to LOOP if no parts needed for subsystem

Line 22 branch to LOOP if parts already found for subsytem

Line 23 if component damage of job is greater than the serviceable percentage of parts in the potential source vehicle, branch to LOOP

Line 24 draw a random comparitor

Line 25 if random comparitor is larger than proportion of serviceable parts, branch to LOOP

Line 26 parts found, record entity number of source vehicle

Line 27 add 1 to number of source vehicles found, mark parts as found by adding 1 to element of DAM.REC array pertaining to source vehicle

Line 28 end of search loop for one potential source vehicle

Line 29 check to see if number of source vehicles found equals number needed, if so branch to AHEAD

Line 30 end of search loop for vehicles in the WT.QUEUE

Lines 31-43 do search procedure for vehicles in WP.QUEUE

Lines 44-45 after all searches are complete, if not enough source vehicles found, set CAN to zero and return

Lines 46-50 prepare to cannibalize by removing job from the set it is in and placing it in WS.QUEUE

Lines 51-55 cancel evacuations for source vehicles

Lines 56-67 cancel PARTS.COME events for source vehicles and reschedule them either at their original time or the time the new parts will arrive, whichever is later

Lines 68-73 find an empty row of the CAN.REC array and set IN.CAN attribute of the job to the row index

Line 74 store entity numbers of source vehicles in the row of the CAN.REC array

Line 75 release core for temporary storage

18. The SUBSTITUTE Routine

This routine is called by the REPAIR event to actually substitute the parts that were found in the CANNIBAL routine, in order to perform the work. The routine

checks the MISSION attribute of the crew performing the repair and removes the parts only from the corresponding subsystems. Evacuations are rescheduled for those jobs that need them, and the row of the CAN.REC array is zeroed out.

Explanation of the coding:

Lines 2-3 definition of local variables in the routine

Line 4 set entity pointers

Line 5 set temporary variable to workorder number of job

Lines 6-7 determine number of subsystems depending on vehicle type

Lines 8-9 determine subsystems for which substitutions are to be made depending on MISSION attribute of crew

Line 10 loop through appropriate subsystems

Line 12 add the proportion of damaged parts for the job to those of the source vehicle

Line 13-14 reduce number of jobs to which source vehicle will supply parts and check to see if the number is zero, if not branch to ZERO

Lines 15-18 if it is zero, and source vehicle is in WT.QUEUE reschedule an evacuation

Line 19-20 zero out the element of the CAN.REC array

Line 21 end of loop

19. The DET.ALLOC Routine

In each BATTLE event, the DET.ALLOC routine is called to determine which, if any, of the forward

detachments are to be attacked by the enemy. A probability of detection is computed for each forward detachment as a function of its distance from the fighting and the number of vehicles present at the unit. The ATTACK routine is then called for each detected unit.

Explanation of the coding:

Lines 2-4 definition of local variables for the routine

Line 5 loop through each forward detachment

Lines 6-7 compute probability of detection for the unit

Line 8 calculate number of personnel present at the unit

Lines 9-10 if unit already destroyed print message and branch to LOOP

Line 11 draw random comparitor

Lines 12-16 if comparitor is less than the probability of detection print message, call ATTACK routine, calculate number of personnel killed in attack, print message

Lines 17-19 if comparitor is larger than probability, print message that unit was not detected, repeat for all units

20. <u>The ATTACK Routine</u>

The ATTACK routine is called by the DET.ALLOC routine each time a forward detachment is detected by the for the duration of the attack. The repairmen left alive at the end of the attack are then reorganized into crews, and work is resumed.

Explanation of coding:

Lines 2-5 definition of local variables in the routine

Line 6 set MAINT.UNIT entity pointer

Line 7 evaluate attrition of inspector and replace him if he is killed

Lines 9-15 evaluate attrition for each individual in the unit

Lines 16-19 calculate number of crews of people left alive

Lines 20-21 determine if there is an odd repairman of either type

Line 22 loop through all crews in maintenance unit that were busy prior to attack

Line 23-25 update number of personnel alive in automotive crews until supply of live repairmen is exhausted

Lines 26-32 when supply of live repairmen is exhausted, set the OCCUPATION attribute of the remaining crews to DEAD and cancel the REPAIR events for jobs they are working on

Lines 33-41 do sequence of lines 23-32 for armament crews

Lines 42-50 if any live repairmen are left assign them to idle crews

Lines 50-58 if there is an odd number of either type of repairmen assign him to a crew

Lines 59-64 cancel all REPAIR events and reschedule them after the attack is over, set flag attribute (LOOP.CH) to 1

Lines 65-66 set flag attribute back to 0

Lines 67-72 cancel all DIAGNOSIS events and reschedule them after the attack is over, set flag attribute to 1

Lines 73-74 set flag attribute back to 0

Lines 75-76 set entity pointers

Lines 77-116 if no more automotive repairmen at unit, evacuate all jobs at the unit with mobility damage, cancel all repairs of automotive jobs, cancel PARTS.COME events for these jobs, cancel DIAGNOSIS events for these jobs

Lines 117-156 do same sequence as lines 77-116 for armament jobs if there are no more armament repairmen

21. <u>The COMP.TIMES Routine</u>

The purpose of this routine is to convert the time related values that it reads as data into beta probability distribution parameters that can be used with the SIMSCRIPT beta random number generator to generate random times to completion for the various maintenance functions. The beta parameters are stored in the A array; and the time values in the form of the minimum, expected, and maximum times for completion of the activity, are stored in the T.ACTION array.

Explanation of the coding:

Lines 2-3 definition of local variables in the routine

Line 4 print heading

Lines 5-10 compute beta parameters

Lines 11-12 print values

22. <u>The INIT.PRINT Routine</u>

This routine is used to print the data values supplied by the user as inputs. They include parameters that govern the maintenance process, as well as the battle and recovery operations. Since the routine is composed of print statements and format specification statements, no explanation of the coding is given.

23. <u>The GAMMA.F Routine</u>

In generating beta distributed random numbers, SIMSCRIPT uses its internal gamma random number generating routine. Some of the parameters that are generated for use in the gamma routine are such that an error is produced. Consequently this routine, which is more robust, was suggested by the designers of SIMSCRCRIPT. This routine overrides the internal GAMMA.F routine. No explanation of the coding is given.

D. VARIABLE DEFINITIONS

1. <u>Input Variables</u>

The values of the following list of variables are supplied by the user. SIMSCRIPT reads data in free format, and it is only important that the values be input in the correct order, and that the data be consistant with their definition in the program in terms of integer or real mode. These variables are listed in the order that they are input.

REP.COUNT - integer, number of replications desired

SEED - integer, number that generates array of random number seeds

P.TANK - real, proportion of tanks in the blue force

W.FIGHT - integer, specifies the initial force configuration and has the following possible values:

1 - split brigade configuration with battalions 1 and 2 engaged first

2 - split brigade configuration with battalions 3 and 4 engaged first

3 - combined brigade configuration

X.RAT - real, initial exchange ratio (red/blue)

BZERO - real, initial blue force level

RZERO - real, initial red force level

BP - real, red breakpoint; proportion of red survivors that cause the red force to assume defensive posture

R2.ECH - real, red second echelon rate of advance (km/hr)

CCSL - real, cross country speed of recovery vehicle when loaded (km/hr)

CCSU - real, cross country speed of recovery vehicle when unloaded (km/hr)

MCPD.ZERO - real, intial distance from the FLOT to the forward detachments (km)

S.ECH - real, distance between red echelons (km)

SELF.LIKE - real, proportion of damaged vehicles that

are self or like vehicle recovered

UNREC - real, proportion of damaged vehicles that are unrecoverable

R.VEHS - real, total number of recovery vehicles in the brigade

TH - real, average hookup time for recovery missions (mins)

LOS.PR - real, probability of line of sight

TGT.PRI - real, priority of recovery vehicles as red targets

PR.INC.ID - real, probability that a recovery vehicle will be incorrectly identified on the battlefield

LEAD.TIME - real, amount of time in simulation before first BATTLE event (hours)

MTTF - real, mean time to failure of blue vehicles (operating hours)

USE.PER - real, proportion of time that the vehicles are operated

D - real, disorientation factor for recovery vehicles; number between 0.0 and 1.0

COD - real, initial distance from the maintenance company to the FLOT (km)

ALFA - real, shaping factor for probability of detection formula

CON.SPEED - real, convoy speed for movement of maintenance units (km/hr)

SETUP.TIME - real, time needed by maintenance unit to resume maintenance activities once new position is reached (minutes)

B.DIST - real, distance from the FLOT to detachment that causes detachment to move to a different position (km)

PK.PERS - real, probability that an individual soldier will become a casualty when forward detachment is attacked

NMC.FOLKS - real, initial number of automotive repairmen at the maintenance company

NFC.FOLKS - real, initial number of armament repairmen at the maintenance company

V.CO.INIT - real, number of vehicles owned by maintenance company

N.FWD.DET - integer, number of forward detachments

NMF.FOLKS - real, initial number of automotive repairmen at each forward detachment

NFF.FOLKS - real, initial number of armament repairmen at each forward detachment

V.FS.INIT - real, number of vehicles owned by the forward detachments

N.BNS - integer, number of supported battalions

P.MOB - real, proportion of system failures that are mobility related

P.FWD.FIX - real, percent of damage that can be

repaired at a forward detachment

PR.HAVE.PARTS.FWD - real, probability that parts are available at a forward detachment

PR.REAR.HAVE.PARTS - real, probability that parts are available at the maintenance company

P.CO.FIX - real, percent of damage that can be repaired at the maintenance company

T.ACTION - 2 dimensional (8 by 3) real array, 8 sets of 3 time values for the following activities:

1 - inspection time at forward detachment

2 - inspection time at maintenance company

3 - repair time for automotive jobs

4 - repair time for armament jobs

5 - waiting time for repair parts delivery

6 - movement time from forward detachment to company

7 - waiting time for evacuation from forward detachment to company

8 - waiting time for evacuation from company to division rear

2. <u>Global Variables</u>

The following variables are globally defined in the preamble, and therefore have the same value throughout the program.

A - real, 2 dimensional array, values of the beta

parameters

ARM.REP.TIME - real, temporary location of the repair times for armament jobs, TALLY statement computes mean of this variable

ATT.CONST - real, relates exchange ratio to number of blue systems alive

B.ALIVE - real, number of blue systems alive

BAT.NUM - integer, counter for number of battle sequences

B1.ALIVE - real, number of blue systems alive in team 1

B2.ALIVE - real, number of blue systems alive in team 2

CAN.REC - integer, 2 dimensional array, stores the job entity numbers of source vehicles for cannibalization

CAS.COUNT - real, number of blue casualties in a battle, TALLY statement computes sum of the values of this variable

COUNT - integer, battle sequence counter used to initiate divisional echelon spacing

DAM.REC - real, 2 dimensional array, stores component damage percentages for all jobs

DOWN.TIME - real, repair cycle time for individual jobs, a TALLY statement computes the mean of this variable

EX.RAT - real, exchange ratio (red/blue)

FLOT.MOVE - real, rate of movement of the FLOT during battle (km/hr)

LIGHT.STAT - real, records whether day or night conditions exist, has the following possible values

DAY - defined to mean 1.0

NIGHT - defined to mean 0.5

LS - real, probability of line of sight

M.REP.TIME - real, temporary location of the repair times for automotive jobs; TALLY statement computes the mean of this variable

MCPD - real, distance from forward detachment to FLOT (km)

MCPD1 - distance from detachments in team 1 to FLOT (km)

MCPD2 - distance from detachments in team 2 to FLOT (km)

NUM.EVAC.REAR - integer, number of vehicles evacuated out of the brigade area

NUM.RETURN.BATTLE - integer, number of vehicles repaired and returned to the fighting force

QUIT - integer, number of system failures since last BATTLE event

QUIT1 - integer, number of system failures in team 1 since last BATTLE event

QUIT2 - integer, number of system failures in team 2 since last BATTLE event

R.CAS.COUNT - real, number of red casualties in a battle, TALLY statement computes the sum of the values

of this variable

REC.NUM - real, number of recovery vehicles alive

REC1.NUM - real, number of recovery vehicles alive in team 1

REC2.NUM - real, number of recovery vehicles alive in team 2

S.CAS - real sum of CAS.COUNT from TALLY statement

S.R.CAS - real, sum of R.CAS.COUNT from TALLY statement

SPACE.ECH - real, red echelon spacing (km)

SUM.NEED.REC - real, sum of number of vehicles that need recovery in each battle; computed by TALLY statement from sum of TOT.NEED.REC

SUM.REC - real, tallied sum of TOT.REC variable

T.PARTS.COME - real, time a job waits for parts: TALLY statement computes mean of this variable

TI.TIME - real, time required to perform initial inspection; TALLY statement computes the mean of this variable

TOT.FOLKS - real, total number of maintenance personnel in the brigade area at the start of the simulation

TOT.NEED.REC - integer, number of vehicles needing recovery in a battle; TALLY statement computes the sum of the values of this variable

TOT.REC - integer, number of vehicles recovered in a battle; TALLY statement computes the sum of this variable

TRY - integer, replication counter

WHO.FIGHT - integer, same as W.FIGHT, specifies which team is engaged in battle

WORK.ORDER - integer, running counter of jobs entering maintenance system

APPENDIX D COMPUTER OUTPUT

```
INPUT TIME PARAMETERS

   I=1  A(1)=   2.87  A(2)=   4.12  T.A(1)=  .50  T.A(2)= 1.50  T.A(3)= 2.60
   I=2  A(1)=   2.87  A(2)=   4.12  T.A(1)=  .50  T.A(2)= 1.50  T.A(3)= 2.60
   I=3  A(1)=   1.09  A(2)=   4.63  T.A(1)=  .60  T.A(2)=11.63  T.A(3)=51.30
   I=4  A(1)=    .90  A(2)=   4.57  T.A(1)=  .10  T.A(2)= 8.80  T.A(3)=45.80
   I=5  A(1)=    .19  A(2)=   4.09  T.A(1)= 6.00  T.A(2)=59.00  T.A(3)=+E 02
   I=6  A(1)=   1.94  A(2)=   4.62  T.A(1)= 6.00  T.A(2)=12.00  T.A(3)=24.00
   I=7  A(1)=   2.29  A(2)=   4.50  T.A(1)= 2.30  T.A(2)= 4.60  T.A(3)= 8.30
   I=8  A(1)=   2.94  A(2)=   4.05  T.A(1)= 2.30  T.A(2)= 4.53  T.A(3)= 6.80

INPUT PARAMETERS

   THE PROPORTION OF JOBS THAT ARE TANKS IS .4900
        THE REST ARE APC'S
   THE PROPORTION OF SYSTEM FAILURES THAT ARE AUTOMOTIVE IS .7330
        THE REST ARE ARMAMENT JOBS
   THERE ARE 4 SUPPORTED BATTALIONS IN THE BRIGADE
   THE PERCENTAGE OF DAMAGE THAT CAN BE FIXED FORWARD IS .6000
   THE PROBABILITY THAT THE FORWARD DET. WILL HAVE THE PARTS IS .5000
   THE PERCENTAGE OF DAMAGE THAT CAN BE FIXED AT THE COMPANY IS .7500
   THE PROBABILITY THAT THE COMPANY HAS THE PARTS IS .7000

BATTLE AND RECOVERY INPUT PARAMETERS

   EXCHANGE RATIO AT BEGINNING OF BATTLE 7.000
   BLUE FORCE LEVEL AT START OF BATTLE  142.
   RED FORCE LEVEL AT START OF BATTLE  176.
   RED BREAK POINT IS  .60 SURVIVING
   RED SECOND ECHELON ADVANCE RATE  3.0 KM/HR
   ECHELCN SPACING  5. KM
   RECOVERY VEHICLE TARGET PRIORITY  5
   PROBABILITY OF INCORRECT IDENTIFICATION OF RECOVERY VEHICLE  .10
   PROBABILITY OF LINE OF SIGHT .30
   NUMBER OF RECOVERY VEHICLES AT START  30
   PROPORTION OF SELF AND LIKE RECOVERY  .07
   PROPORTION OF UNRECOVERABLE VEHICLES  .10
   EXPECTED HOOKUP TIME    .5 HOURS
   DISTANCE FROM BATTLE SITE TO MCP AT START OF BATTLE   5 KM
   CROSS COUNTRY SPEED LOADED 30.0 KM/HR
   CROSS COUNTRY SPEED UNLCADED 60.0 KM/HR
   MEAN TIME BETWEEN SYSTEM FAILURES  200.0 OPERATING HOURS
   WARNING TIME BEFORE START OF BATTLE  72 HOURS
   TIME FOR SETUP AFTER MOVE   15 MINUTES
   CONVOY SPEED DURING MOVE 20.00 KM/HR
   BREAKPOINT DISTANCE AT WHICH DET. MOVES   2 KM
   PROBABILITY OF PERSONNEL CASUALTY IN ATTACK   .30
```

JOB LIST

WO	TD	V	U	H	FP	MD					DAM.REC						
1	.32	2	2	0	0.	.04	0.	0.	.27	0.	0.	0.	0.	0.	0.		
2	.37	1	2	0	0.	.17	0.	0.	.12	0.	0.	0.	0.	0.	0.	0.	0.
3	.47	2	3	0	0.	.11	.57	0.	0.	0.	0.	0.	0.	0.	0.		
4	.65	1	2	0	.17	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	.38
5	.70	1	1	0	0.	.07	0.	.64	0.	0.	0.	0.	0.	0.	0.	0.	0.
6	.71	2	4	0	.04	0.	0.	0.	0.	0.	0.	0.	0.	.50	0.		
7	.75	1	4	0	0.	.12	0.	0.	0.	0.	0.	.44	0.	0.	0.	0.	0.
8	.79	1	3	0	0.	.08	0.	0.	0.	0.	0.	.40	0.	0.	0.	0.	0.
9	.81	1	3	0	0.	.18	0.	0.	0.	0.	.02	0.	0.	0.	0.	0.	0.
10	.91	2	4	0	.19	0.	0.	0.	0.	0.	0.	0.	0.	0.	.65		
11	1.05	1	1	0	0.	.06	0.	0.	0.	.61	0.	0.	0.	0.	0.	0.	0.
12	1.05	1	1	0	0.	.15	0.	0.	.13	0.	0.	0.	0.	0.	0.	0.	0.
13	1.41	1	2	0	0.	.02	.64	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
14	1.53	2	1	0	0.	.09	0.	0.	0.	0.	.39	0.	0.	0.	0.		
15	1.82	2	3	0	0.	.14	0.	0.	.30	0.	0.	0.	0.	0.	0.		
16	1.86	1	2	0	0.	.11	0.	0.	0.	0.	0.	.13	0.	0.	0.	0.	0.
17	2.01	2	1	0	0.	.17	0.	0.	0.	.48	0.	0.	0.	0.	0.		
18	2.25	1	1	0	0.	.13	0.	0.	.92	0.	0.	0.	0.	0.	0.	0.	0.
19	2.33	1	2	0	0.	.18	0.	0.	.83	0.	0.	0.	0.	0.	0.	0.	0.
20	2.37	1	2	0	.01	0.	0.	0.	0.	0.	0.	0.	.72	0.	0.	0.	0.
21	2.41	1	1	0	0.	.13	0.	0.	.42	0.	0.	0.	0.	0.	0.	0.	0.
22	2.43	2	4	0	0.	.08	.64	0.	0.	0.	0.	0.	0.	0.	0.		
23	2.56	2	1	0	.08	0.	0.	0.	0.	0.	0.	0.	0.	.56	0.		
24	2.77	1	1	0	.14	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	.09
25	2.80	1	4	0	.04	0.	0.	0.	0.	0.	0.	0.	.67	0.	0.	0.	0.

RESULTS OF BATTLE 1

MAINT UNIT # 1 NOT DETECTED P.DET = .1299 D.FLOT = 5.000 VEH = 6.
20 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 2 NOT DETECTED P.DET = .0558 D.FLOT = 5.000 VEH = 3.
20 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 3 NOT DETECTED P.DET = .0558 D.FLOT = 5.000 VEH = 3.
20 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 4 NOT DETECTED P.DET = .0558 D.FLOT = 5.000 VEH = 3.
20 PERSONNEL PRESENT AT UNIT

WHO.FIGHT IS 1 TIME OF BATTLE IS .088 HOURS

NUMBER RECOVERY VEHICLES KILLED THIS BATTLE: 1

RE. VEHS. LEFT: 14 B.ALIVE: 123 R.ALIVE: 194

JOB LIST

WO	TD	V	U	H	FP	MD					DAM.REC						
26	3.01	1	2	1	.17	.02	0.	.50	0.	0.	0.	0.	0.	.32	0.	0.	0.
27	3.01	1	1	1	.79	.47	.63	.89	.87	0.	0.	0.	.70	0.	.96	.43	.15

```
 NUMBER RECOVERY VEHICLES KILLED THIS BATTLE:   2
 # RE. VEHS. LEFT: 12  B.ALIVE: 109  R.ALIVE: 210

JOB LIST
 WO  TD   V U H  FP   MD                         DAM.REC
 47 3.33 2 2 2  .13  .11 0.  .02 0.  0.  0.  0.  0.  0.  .84
 48 3.34 2 1 1  .15  .77 .92 .23 0.  .77 .12 .57 0.  0.  .39
 49 3.36 2 2 1  .86  .92 .59 .64 .39 .33 .12 .22 .55 .04 .55
 50 3.36 1 1 1  .80  .41 .12 0.  0.  0.  .33 0.  .43 .25 .49 0.  .56
 51 3.36 2 2 1  .77  .47 .53 0.  0.  .84 0.  .01 .94 .16 0.
 52 3.38 1 1 1  .12  .03 0.  0.  0.  0.  0.  .84 .03 0.  0.  0.  0.
 53 3.38 2 1 1  .68  .61 0.  .58 .81 0.  .23 .73 0.  .83 .28
 54 3.38 2 2 1  .52  .72 .86 .06 0.  0.  .81 .59 .61 .08 0.
 55 3.40 2 2 1  .54  .79 .89 .44 .90 0.  .40 .00 .65 .09 0.
 56 3.40 1 2 1  .18  .21 0.  .15 0.  0.  0.  0.  0.  .76 0.  0.  0.
 57 3.42 1 2 0 0.   .00 0.  0.  0.  0.  .68 0.  0.  0.  0.  0.  0.
 58 3.42 1 2 1  .43  .89 .51 .85 0.  .96 .72 .38 0.  0.  0.  .45 .96
RESULTS OF BATTLE  4
MAINT UNIT # 1 NOT DETECTED    P.DET =  .0710  D.FLOT =  7.000 VEH =   7.
       14 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 2 NOT DETECTED    P.DET =  .1212  D.FLOT =  7.000 VEH =  11.
       20 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 3 NOT DETECTED    P.DET =  .0313  D.FLOT =  6.000 VEH =   3.
       20 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 4 NOT DETECTED    P.DET =  .1035  D.FLOT =  6.000 VEH =   7.
       20 PERSONNEL PRESENT AT UNIT
 WHO.FIGHT IS 2   TIME OF BATTLE IS  .127 HOURS
 NUMBER RECOVERY VEHICLES KILLED THIS BATTLE:   1
 # RE. VEHS. LEFT: 13  B.ALIVE: 113  R.ALIVE: 214

JOB LIST
 WO  TD   V U H  FP   MD                         DAM.REC
 59 3.50 1 3 1  .65  .25 0.  .36 0.  0.  0.  0.  .47 0.  .16 0.  .06
 60 3.52 1 3 1  .79  .16 .60 0.  0.  0.  0.  0.  .49 .74 0.  .57 .26
 61 3.58 2 3 1  .90  .75 .08 .13 0.  .54 .20 .48 .16 .22 .99
 62 3.60 2 3 1  .41  .69 .84 0.  .93 .22 0.  .36 0.  0.  .79
 63 3.64 1 3 0 0.   .05 0.  0.  .71 0.  0.  0.  0.  0.  0.  0.  0.
 64 3.66 2 3 1  .89  .99 .39 .14 .55 .64 .42 .52 .77 .61 .74
 65 3.69 1 4 2  .84  .06 0.  0.  0.  0.  .63 0.  .84 .48 0.  .03 .60
 66 3.72 1 3 1  .30  .93 .79 .51 .02 .73 .41 .95 .18 .52 0.  0.  0.
 67 3.72 1 4 1  .21  .78 .24 .56 .50 .63 .99 0.  0.  0.  0.  .68 0.
 68 3.74 1 3 1  .90  .87 .82 .20 .10 0.  .29 .06 .22 .33 .37 .10 0.
```

```
69 3.76 2 4 1 .67 .03 0. 0. 0. 0. 0. .94 .16 .71 0.
70 3.79 1 3 1 .38 .57 0. 0. 0. .29 .64 .11 0. .75 .99 0. 0.
71 3.81 1 3 1 .71 .81 0. .52 .82 .09 .75 .96 .60 0. .04 .14 .15
RESULTS OF BATTLE  5
MAINT UNIT # 1 NOT DETECTED    P.DET =  .0302  D.FLJT =  7.000 VEH =   4.
      14 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 2 NOT DETECTED    P.DET =  .0842  D.FLOT =  7.000 VEH =   8.
      20 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 3 NOT DETECTED    P.DET =  .1093  D.FLOT =  7.000 VEH =  10.
      20 PERSCNNEL PRESENT AT JNIT
MAINT UNIT # 4 DETECTED THIS BATTLE    20 PERSONNEL PRESENT
       5 PEOPLE KILLED IN THIS ATTACK
   WHO.FIGHT IS 1   TIME OF BATTLE IS  .619 HOURS
   NUMBER RECOVERY VEHICLES KILLED THIS BATTLE:   5
   # RE. VEHS. LEFT:  7  B.ALIVE:  69  R.ALIVE: 217

JOB LIST
 WO  TD   V U H  FP   MD                      DAM.REC
72 3.88 2 2 1 .92 .06 0. 0. 0. 0. 0. .74 .73 .60 .14
73 3.90 1 2 1 .38 .67 .74 .21 0. .40 .75 0. 0. 0. 0. .12 .49
74 3.91 1 2 2 .38 .19 0. .48 0. 0. 0. 0. 0. 0. 0. .50 .98
75 3.91 2 1 1 .66 .56 0. 0. .35 .08 0. .13 .92 .98 0.
76 3.91 2 1 2 .12 .15 0. .92 0. 0. 0. 0. .98 0. 0.
77 3.91 2 1 1 .79 .32 0. 0. .92 .71 0. 0. .91 0. .90
78 3.92 2 1 1 .36 .06 0. 0. 0. .43 0. 0. 0. 0. .80
79 3.94 2 2 0 0. .14 0. 0. .38 0. 0. 0. 0. 0. 0.
80 3.94 1 1 1 .88 .18 .27 0. 0. 0. 0. 0. .72 1.03 .46 .15 0.
81 3.95 1 1 1 .66 .05 0. .51 0. 0. 0. 0. 0. .06 0. .24 .49
82 3.96 2 1 1 .19 .94 .54 .88 .63 .08 .04 .72 0. .83 0.
83 3.96 2 2 1 .65 .30 0. .57 0. .03 0. 0. .49 0. .83
84 3.96 2 1 1 .67 .92 .17 .78 .87 .73 .57 .53 0. .47 .51
85 3.96 1 1 1 .62 .89 0. .23 .19 .02 .44 .92 .35 .20 0. .92 0.
86 3.97 1 2 2 .77 .04 0. 0. .04 0. 0. 0. 0. .21 .57 .52 .32
87 3.97 1 1 1 .21 .48 .77 0. .64 0. .01 0. 0. 0. .11 0. 0.
88 3.97 2 1 1 .73 .99 .89 .97 .57 .72 .00 .61 .68 0. .18
89 3.97 1 2 1 .35 .73 0. .85 0. .53 .56 .49 0. .58 0. .39 0.
90 3.98 1 1 1 .31 .69 .38 .52 0. .75 0. .64 .91 .59 0. 0. 0.
91 3.98 2 2 1 .22 .65 .72 0. 0. .25 .18 .25 .07 0. 0.
92 3.99 2 2 0 0. .18 .88 0. 0. 0. 0. 0. 0. 0. 0.
93 3.99 2 2 1 1.00 .16 0. 0. .74 0. 0. 0. .72 .33 .40
94 3.99 1 2 1 .80 .65 .07 0. .16 .33 .96 0. .82 .41 .26 .64 0.
95 4.01 2 1 1 .19 .39 .37 0. 0. .35 0. 0. .66 0. 0.
96 4.01 1 2 1 .26 .98 .64 .04 .40 .27 .09 .30 0. 0. 0. 0. .30
97 4.02 2 2 1 .39 .69 .87 0. .20 .88 0. .08 0. .29 0.
98 4.02 1 2 1 .31 .58 .05 0. 0. 0. .71 .20 .42 0. .53 0. 0.
RESULTS OF BATTLE  6
```

```
MAINT UNIT # 1 DETECTED THIS BATTLE     14 PERSONNEL PRESENT
       2 PEOPLE KILLED IN THIS ATTACK
MAINT UNIT # 2 DETECTED THIS BATTLE     20 PERSONNEL PRESENT
       5 PEOPLE KILLED IN THIS ATTACK
MAINT UNIT # 3 NOT DETECTED     P.DET =  .0842  D.FLOT =  7.000 VEH =   8.
      20 PERSCNNEL PRESENT AT JNIT
MAINT UNIT # 4 NOT DETECTED     P.DET =  .0302  D.FLOT =  7.000 VEH =   4.
      15 PERSONNEL PRESENT AT JNIT

   WHO.FIGHT IS 3   TIME OF BATTLE IS  .038 HOURS

   NUMBER RECOVERY VEHICLES KILLED THIS BATTLE:   1

   # RE. VEHS. LEFT: 19  B.ALIVE: 178  R.ALIVE: 218

JOB LIST

 WO  TD   V U H  FP   MD                        DAM.REC

 99 4.12 1 3 1  .78  .26 0.   .92  .50 0.   0.   0.   0.   .77  .21  .63  .46
100 4.15 1 4 1  .17  .08  .98 0.   0.   0.   0.   0.   0.   .44 0.   0.   0.
101 4.15 1 3 1  .85  .59  .18  .28 0.   .23 0.   .52  .79  .31 0.   .74  .97
102 4.17 2 2 0 0.    .02 0.   0.   0.   0.   .95 0.   0.   0.   0.
103 4.18 2 3 1  .58  .76  .35  .34  .32  .94 0.   .42 0.   .05  .92
104 4.18 1 1 1  .88  .75  .60 0.   0.   .30  .46  .77  .23  .37 0.   .41  .63
RESULTS OF BATTLE  7

MAINT UNIT # 1 DETECTED THIS BATTLE     12 PERSONNEL PRESENT
       1 PEOPLE KILLED IN THIS ATTACK
MAINT UNIT # 2 NOT DETECTED     P.DET =  .4609  D.FLOT =  3.000 VEH =  15.
      15 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 3 NOT DETECTED     P.DET =  .0842  D.FLOT =  7.000 VEH =   8.
      20 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 4 NOT DETECTED     P.DET =  .0437  D.FLOT =  7.000 VEH =   5.
      15 PERSONNEL PRESENT AT UNIT

   WHO.FIGHT IS 3   TIME OF BATTLE IS  .042 HOURS

   NUMBER RECOVERY VEHICLES KILLED THIS BATTLE:   1

   # RE. VEHS. LEFT: 18  B.ALIVE: 172  R.ALIVE: 219

JOB LIST

 WO  TD   V U H  FP   MD                        DAM.REC

105 4.27 1 2 0  .06 0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   .62 0.
106 4.29 2 1 1  .98  .96  .31  .62  .56  .27  .89  .59  .88  .76  .52
107 4.32 2 2 2  .53  .13 0.   0.   0.   .67 0.   0.   .08  .40 0.
108 4.32 2 3 1  .92  .93  .39  .54  .14  .06  .64  .71  .02  .86  .47
109 4.32 1 2 1  .04  .72 0.   .54  .37  .89 0.   .31 0.   .50 0.   0.   0.
```

```
110 4.32 1 1 1  .15  .55 0.      .69  .85  .49 0.   0.   0.    0.  0.   0.   .54
111 4.33 2 2 1  .02  .40    .54 0.    .27 0.    0.   0.     .34 0.  0.
RESULTS OF BATTLE  8

MAINT UNIT # 1 DETECTED THIS BATTLE     11 PERSONNEL PRESENT
      2 PEOPLE KILLED IN THIS ATTACK
MAINT UNIT # 2 NOT DETECTED    P.DET =  .5025  D.FLOT =  3.000 VEH =  19.
      15 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 3 NOT DETECTED    P.DET =  .0710  D.FLOT =  7.000 VEH =   7.
      20 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 4 DETECTED THIS BATTLE     15 PERSONNEL PRESENT
      1 PEOPLE KILLED IN THIS ATTACK

   WHO.FIGHT IS 3   TIME OF BATTLE IS  .049 HOURS

   NUMBER RECOVERY VEHICLES KILLED THIS BATTLE:   0

   # RE. VEHS. LEFT: 18  B.ALIVE: 164  R.ALIVE: 219

JOB LIST
 WO  TD  V U H  FP   MD                      DAM.REC

112 4.44 1 3 1  .42  .97  .79  .36  .31  .19  .58  .24  .12  .89 0.   0.   0.
113 4.46 2 1 1  .28  .81  .39  .21 0.    .81  .89  .29 0.    .23 0.
114 4.46 2 3 1  .48  .23 0.    .66 0.   0.   0.   0.   0.   0.   .40
115 4.60 2 3 1  .49  .62 0.    .23  .24  .72  .70 0.   0.   0.   .18
116 4.62 1 3 1  .86  .71  .96  .38 0.    .72  .99 0.    .15  .71 0.    .06  .71
117 4.65 2 2 2  .94  .06 0.   0.   0.   0.    .24 0.    .16  .14  .77
118 4.74 1 3 1  .30  .35 0.    .49 0.   0.    .21 0.   0.   0.   0.    .14 0.
119 4.78 1 1 1  .05  .19  .74 0.   0.   0.   0.   0.    .24 0.   0.   0.   0.
RESULTS OF BATTLE  9

MAINT UNIT # 1 NOT DETECTED    P.DET =  .4206  D.FLOT =  3.000 VEH =  12.
      9 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 2 DETECTED THIS BATTLE     15 PERSONNEL PRESENT
      1 PEOPLE KILLED IN THIS ATTACK
MAINT UNIT # 3 DETECTED THIS BATTLE     20 PERSONNEL PRESENT
      6 PEOPLE KILLED IN THIS ATTACK
MAINT UNIT # 4 NOT DETECTED    P.DET =  .0437  D.FLOT =  7.000 VEH =   5.
      14 PERSONNEL PRESENT AT UNIT

   WHO.FIGHT IS 3   TIME OF BATTLE IS  .101 HOURS

   NUMBER RECOVERY VEHICLES KILLED THIS BATTLE:   3

   # RE. VEHS. LEFT: 15  B.ALIVE: 156  R.ALIVE: 220

JOB LIST
 WO  TD  V U H  FP   MD                      DAM.REC
```

```
120 4.88 1 3 1  .09  .28  .88 0.   0.   .82 0.   0.   .67 0.   0.   0.   0.
121 4.90 2 1 1  .17  .87  .99  .56  .43  .76 0.   .70 0.   0.   .69
122 4.91 1 4 0  .13 0.   0.   0.   0.   0.   0.   0.   0.   0.   .36 0.   0.
123 4.91 1 4 1  .50  .92  .33  .83  .98  .19  .01  .18  .33 0.   0.   .79  .99
124 4.92 2 2 1  .65  .86 0.    .83  .27  .74  .20  .55  .91 0.   .68
125 4.93 1 4 1  .69  .99  .15  .05  .45  .37  .35  .35 0.   0.   .71  .44  .74
126 4.94 2 1 1  .22  .91  .95  .02 0.    .01  .33  .60 0.   .74 0.
127 4.95 2 2 1  .41  .21 0.   0.    .68 0.   0.   0.   .26 0.   0.
128 4.95 2 3 1  .65  .68  .13 0.    .46  .27  .30 0.   .79 0.   .96
129 4.96 1 1 1  .60  .84 0.    .85  .14  .40  .00  .51  .19 0.   .82  .16 0.
130 4.96 1 2 1  .49  .75  .26  .61  .05  .51 0.   0.   .16 0.   0.   0.   .09
131 4.97 1 3 2  .27  .18 0.    .79 0.   0.   0.   0.   .02 0.   0.   0.   0.
```

RESULTS OF BATTLE 10

```
MAINT UNIT # 1 NOT DETECTED    P.DET =  .4352  D.FLOT =  3.000 VEH =  13.
         9 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 2 NOT DETECTED    P.DET =  .4352  D.FLOT =  3.000 VEH =  13.
        14 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 3 NOT DETECTED    P.DET =  .0574  D.FLOT =  7.000 VEH =   6.
        14 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 4 NOT DETECTED    P.DET =  .0710  D.FLOT =  7.000 VEH =   7.
        14 PERSONNEL PRESENT AT UNIT
```

WHO.FIGHT IS 3   TIME OF BATTLE IS .867 HOURS

NUMBER RECOVERY VEHICLES KILLED THIS BATTLE: 2

RE. VEHS. LEFT: 13 B.ALIVE: 147 R.ALIVE: 220

JOB LIST

```
 WO  TO   V U H  FP   MD                               DAM.REC

132 5.07 2 1 1  .23  .82  .19  .58  .44 0.    .73  .03 0.    .39 0.
133 5.09 1 4 1  .68  .41  .72 0.   0.   0.    .76 0.   0.    .92  .06 0.    .39
134 5.11 2 2 1  .31  .24 0.   0.   0.    .06 0.   0.    .98 0.   0.
135 5.11 2 4 1  .12  .62 0.   0.    .14  .39  .10  .12 0.   0.    .50
136 5.11 1 3 1  .87  .97  .48  .06  .43  .97  .70  .73  .83 0.    .20  .18  .81
137 5.15 1 3 0 0.    .13 0.   0.   0.   0.   0.    .07 0.   0.   0.   0.   0.
138 5.17 2 3 1  .34  .28 0.    .41 0.   0.   0.    .33 0.    .51 0.
139 5.17 1 4 2  .06  .03 0.   0.    .43 0.   0.   0.   0.    .11 0.   0.   0.
```

RESULTS OF BATTLE 11

```
MAINT UNIT # 1 NOT DETECTED    P.DET =  .3873  D.FLOT =  3.000 VEH =  10.
         9 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 2 NOT DETECTED    P.DET =  .4047  D.FLOT =  3.000 VEH =  11.
        14 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 3 NOT DETECTED    P.DET =  .0842  D.FLOT =  7.000 VEH =   8.
        14 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 4 NOT DETECTED    P.DET =  .0842  D.FLOT =  7.000 VEH =   8.
        14 PERSONNEL PRESENT AT UNIT
```

```
   WHO.FIGHT IS 3   TIME OF BATTLE IS  .079 HOURS
   NUMBER RECOVERY VEHICLES KILLED THIS BATTLE:   1
   # RE. VEHS. LEFT: 12  B.ALIVE: 139  R.ALIVE: 220

JOB LIST
 WO  TD  V U H  FP   MD                          DAM.REC
140 5.26 2 2 1  .43  .23 0.  0.  .49 0.  0.  0.  0.  0.  .22
141 5.27 1 2 1  .59  .75 .33 0.  .32 0.  .35 .96 0.  .10 .96 0.  .47
142 5.28 1 2 1  .75  .90 .75 .38 0.  .40 .05 .37 .17 .41 .67 0.  .99
143 5.28 1 1 1  .21  .06 0.  0.  0.  .28 0.  0.  .14 0.  0.  0.  0.
144 5.30 2 4 1  .06  .35 0.  .20 0.  0.  0.  .75 .80 0.  0.
145 5.31 1 1 2  .92  .17 .37 0.  0.  0.  0.  0.  .94 .32 .13 .86 .32
146 5.32 1 3 1  .75  .29 0.  .87 0.  0.  0.  .22 .93 .42 0.  .42 .33
147 5.33 1 3 1  .96  .17 .75 0.  0.  0.  0.  0.  .07 .57 .18 .82 .41
148 5.35 1 2 1  .34  .09 0.  .42 0.  0.  0.  0.  0.  0.  0.  .74 .47
RESULTS OF BATTLE 12
MAINT UNIT # 1 NOT DETECTED    P.DET =  .3679  D.FLOT =  3.000 VEH =   9.
       9 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 2 DETECTED THIS BATTLE    14 PERSONNEL PRESENT
       3 PEOPLE KILLED IN THIS ATTACK
MAINT UNIT # 3 NOT DETECTED    P.DET =  .0842  D.FLOT =  7.000 VEH =   8.
      14 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 4 NOT DETECTED    P.DET =  .0574  D.FLOT =  7.000 VEH =   6.
      14 PERSONNEL PRESENT AT UNIT
   WHO.FIGHT IS 3   TIME OF BATTLE IS  .099 HOURS
   NUMBER RECOVERY VEHICLES KILLED THIS BATTLE:   0
   # RE. VEHS. LEFT: 12  B.ALIVE: 127  R.ALIVE: 220

JOB LIST
 WO  TD  V U H  FP   MD                          DAM.REC
149 5.47 1 1 1  .33  .99 .75 .65 .42 .58 .73 .70 .84 0.  0.  0.  .62
150 5.49 1 2 1  .14  .02 0.  0.  0.  0.  0.  .46 0.  .74 0.  0.  0.
151 5.51 2 2 1  .17  .26 .99 0.  0.  .76 0.  0.  0.  .44 0.
152 5.53 2 2 1  .22  .49 0.  0.  .80 .08 .32 0.  0.  0.  .20
153 5.54 2 4 1  .41  .78 .04 .20 .14 .50 .62 0.  0.  .67 0.
154 5.56 2 4 1  .54  .07 0.  0.  .49 0.  0.  0.  0.  .31 .50
155 5.62 2 3 1  .93  .67 0.  .38 .35 .35 .37 0.  .45 .41 .92
156 5.68 2 4 1  .38  .88 .94 0.  .21 .46 .56 .62 .02 0.  0.
157 5.70 2 2 2  .29  .10 0.  0.  0.  0.  .88 0.  0.  0.  .91
158 5.76 2 2 1  .78  .68 0.  .03 .96 .32 .95 0.  .41 0.  .60
159 5.78 1 1 1  .49  .35 0.  .83 .75 0.  0.  0.  0.  0.  .48 0.  .45
```

```
160 5.79 2 4 1  .06  .41  .98 0.   0.   0.   0.    .68  .65 0.   0.
RESULTS OF BATTLE 13

MAINT UNIT # 1 NOT DETECTED    P.DET =  .3873  D.FLOT =  3.000 VEH =  10.
       9 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 2 DETECTED THIS BATTLE     11 PERSONNEL PRESENT
       0 PEOPLE KILLED IN THIS ATTACK
MAINT UNIT # 3 NOT DETECTED    P.DET =  .0710  D.FLOT =  7.000 VEH =   7.
      14 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 4 NOT DETECTED    P.DET =  .0842  D.FLOT =  7.000 VEH =   8.
      14 PERSONNEL PRESENT AT UNIT

   WHO.FIGHT IS 3   TIME OF BATTLE IS  .240 HOURS

   NUMBER RECOVERY VEHICLES KILLED THIS BATTLE:   1

   # RE. VEHS. LEFT: 11  B.ALIVE: 109  R.ALIVE: 220

JOB LIST
 WO   TD  V U H  FP   MD                         DAM.REC

161 5.86 2 1 1  .37  .92  .38  .37  .72  .16  .72  .43  .07 0.   0.
162 5.87 1 1 1  .08  .85  .22  .21 0.    .78  .08  .09 0.   0.   0.    .72 0.
163 5.87 1 2 1  .99  .65 0.    .50  .43  .66  .52 0.    .22  .73  .98  .80  .36
164 5.89 1 4 1  .51  .13 0.   0.   0.   0.   0.    .87  .18 0.    .27  .35 0.
165 5.89 2 3 2  .74  .07 0.   0.   0.    .78 0.   0.    .28 0.    .92
166 5.90 1 2 1  .02  .73  .36 0.   0.    .21  .40  .66 0.   0.   0.   0.    .80
167 5.90 1 1 1  .90  .94  .18  .33  .59  .10  .40  .93  .35  .45  .93  .91  .78
168 5.91 1 3 1  .73  .63 0.   0.    .58  .71  .96  .02  .02  .06  .86 0.    .67
169 5.92 1 3 1  .72  .24 0.   0.    .40 0.   0.   0.    .08 0.    .14  .76  .97
170 5.92 2 3 1  .76  .26 0.   0.    .04 0.   0.    .02 0.    .25  .05
171 5.92 2 3 0  .01 0.   0.   0.   0.   0.   0.   0.   0.   0.    .39
172 5.92 1 1 1  .82  .68 0.    .63  .67 0.    .98  .13  .33 0.    .62  .03  .27
173 5.92 1 1 1  .51  .22 0.   0.   0.    .79 0.   0.    .74 0.   0.    .56  .13
174 5.92 1 2 1  .48  .75 0.    .92 0.    .08  .92  .46  .85 0.   0.    .27 0.
175 5.93 1 4 1  .99  .66 0.    .73  .30  .63  .74 0.    .13  .13  .49  .52  .80
176 5.93 2 3 1  .87  .41 0.   0.    .07  .90 0.   0.    .27  .65  .03
177 5.94 2 4 0 0.    .18  .17 0.   0.   0.   0.   0.   0.   0.   0.
178 5.94 2 2 1  .80  .55 0.    .59  .70 0.   0.    .91  .23  .43 0.
179 5.94 1 4 2  .43  .07 0.   0.   0.   0.   0.    .02 0.    .39 0.    .77 0.
180 5.95 2 3 1  .37  .25 0.   0.   0.   0.   0.    .81 0.    .70 0.
RESULTS OF BATTLE 14

MAINT UNIT # 1 NOT DETECTED    P.DET =  .4609  D.FLOT =  3.000 VEH =  15.
       9 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 2 NOT DETECTED    P.DET =  .4609  D.FLOT =  3.000 VEH =  15.
      11 PERSCNNEL PRESENT AT UNIT
MAINT UNIT # 3 NOT DETECTED    P.DET =  .1435  D.FLOT =  7.000 VEH =  13.
      14 PERSCNNEL PRESENT AT UNIT
MAINT UNIT # 4 NOT DETECTED    P.DET =  .1093  D.FLOT =  7.000 VEH =  10.
      14 PERSONNEL PRESENT AT UNIT
```

```
   WHO.FIGHT IS 3   TIME OF BATTLE IS  .211 HOURS
   NUMBER RECOVERY VEHICLES KILLED THIS BATTLE:   1
   # RE. VEHS. LEFT: 10  B.ALIVE:  92  R.ALIVE: 220

JOB LIST
 WO  TD   V U H  FP   MD                            DAM.REC

181 6.02 1 3 1  .78  .71  .34  .67 0.   0.   .29  .32  .81  .36  .25  .45 0.
182 6.02 2 4 1  .61  .71 0.    .80  .96  .17 0.   .47 0.    .03  .36
183 6.02 1 3 1  .89  .57 0.    .40 0.   0.    .86  .46  .94  .63  .27  .38 0.
184 6.03 1 3 1  .86  .98  .82  .38  .58  .62  .02  .48  .04 0.    .59  .81  .56
185 6.05 2 1 1  .15  .28  .51 0.   0.    .57 0.   0.   0.    .74 0.
186 6.06 2 3 1  .06  .67  .56 0.    .93 0.    .48  .10 0.    .12 0.
187 6.06 1 3 1  .71  .60  .20  .10 0.    .02 0.    .13 0.    .33  .95  .54  .60
188 6.07 2 1 1  .23  .92  .93  .81  .74  .02  .86  .89  .04 0.   0.
189 6.08 2 1 0 0.    .08 0.   0.   0.    .64 0.   0.   0.   0.   0.
190 6.08 2 2 1  .17  .24 0.   0.   0.    .18 0.   0.   0.   0.   1.00
191 6.08 2 4 1  .51  .02 0.   0.   0.   0.   0.    .61  .26 0.    .95
192 6.10 2 4 1  .68  .17 0.   0.   0.    .30 0.   0.    .57  .03 0.
193 6.10 2 3 1  .47  .11  .45 0.   0.   0.   0.   0.   0.    .16 0.
194 6.11 2 2 1  .61  .12 0.   0.   0.   0.    .42 0.    .26 0.    .49
195 6.11 1 2 1  .54  .71 0.    .46  .52 0.    .31  .72  .01  .52 0.    .87 0.
196 6.11 1 2 2  .43  .09 0.   0.   0.    .16 0.   0.   0.    .61  .52 0.   0.
197 6.11 2 4 1  .84  .81  .13 1.00  .88  .43 0.    .50  .32  .22  .61
RESULTS OF BATTLE 15

MAINT UNIT # 1 NOT DETECTED    P.DET =  .4609  D.FLOT =  3.000 VEH =  15.
        9 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 2 DETECTED THIS BATTLE    11 PERSONNEL PRESENT
        2 PEOPLE KILLED IN THIS ATTACK
MAINT UNIT # 3 NOT DETECTED    P.DET =  .1921  D.FLOT =  7.000 VEH =  18.
       14 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 4 NOT DETECTED    P.DET =  .1435  D.FLOT =  7.000 VEH =  13.
       14 PERSONNEL PRESENT AT UNIT

   WHO.FIGHT IS 3   TIME OF BATTLE IS  .402 HOURS
   NUMBER RECOVERY VEHICLES KILLED THIS BATTLE:   3
   # RE. VEHS. LEFT:  7  B.ALIVE:  72  R.ALIVE: 220

JOB LIST
 WO  TD   V U H  FP   MD                            DAM.REC

198 6.19 2 4 1  .94  .06 0.   0.    .85 0.   0.   0.    .09  .14  .54
199 6.21 2 3 1  .76  .27 0.    .59 0.   0.    .49 0.    .25  .44 0.
200 6.22 2 3 1  .07  .98  .72  .03  .12  .80  .97  .35 0.    .70 0.
```

```
201 6.22 2 4 2  .86  .09 0.   0.   0.   0.   .96 0.   .76  .65  .67
202 6.23 2 1 1  .78  .90 0.   .27  .37  .44  .61  .49 0.   .02  .45
203 6.24 2 4 1  .79  .49  .41 0.   0.   0.   .68  .75  .47 0.   .67
204 6.25 2 4 1  .53  .63  .62  .28  .41 0.   .98 0.   0.   .54  .40
205 6.25 1 1 1  .60  .51 0.   .20 0.   .04  .67 0.   .87  .99 0.   .77 0.
206 6.26 2 2 1  .31  .71  .39  .01 0.   0.   .53  .97  .44 0.   0.
207 6.26 1 1 1  .50  .88 1.00 .68 0.   .96  .19  .55 0.   0.   .59  .57 0.
208 6.28 2 2 1  .21  .17 0.   0.   0.   .87 0.   0.   0.   .62 0.
209 6.28 1 2 1  .09  .58  .65  .04 0.   0.   .60 0.   .90 0.   0.   0.   0.
210 6.29 1 1 1  .03  .55 0.   1.00 0.   .80 0.   .86 0.   0.   0.   0.   .37
211 6.29 1 1 1  .25  .62 0.   .31  .47  .55 0.   .50 0.   0.   0.   0.   .20
212 6.29 1 1 1  .82  .35 0.   .04 0.   0.   0.   .80  .21 0.   .25  .79  .48
213 6.29 2 2 2  .41  .02 0.   0.   0.   0.   .25 0.   .45 0.   0.
214 6.29 2 3 1  .88  .34 0.   .30 0.   0.   .22 0.   .00  .73  .23
215 6.30 2 4 1  .58  .07 0.   0.   0.   .31 0.   0.   .80  .96 0.
216 6.30 1 4 1  .16  .44  .04  .89 0.   .73 0.   0.   0.   0.   .69 0.   0.
RESULTS OF BATTLE 16

MAINT UNIT # 1 DETECTED THIS BATTLE      9 PERSONNEL PRESENT
      0 PEOPLE KILLED IN THIS ATTACK
MAINT UNIT # 2 DETECTED THIS BATTLE      9 PERSONNEL PRESENT
      2 PEOPLE KILLED IN THIS ATTACK
MAINT UNIT # 3 NOT DETECTED    P.DET =  .2007  D.FLOT =  7.000 VEH =  19.
     14 PERSONNEL PRESENT AT UNIT
MAINT UNIT # 4 NOT DETECTED    P.DET =  .1831  D.FLOT =  7.000 VEH =  17.
     14 PERSONNEL PRESENT AT UNIT

   WHO.FIGHT IS 3   TIME OF BATTLE IS  .548 HOURS

   NUMBER RECOVERY VEHICLES KILLED THIS BATTLE:   6

   # RE. VEHS. LEFT:  1  B.ALIVE:  43  R.ALIVE: 220

BLUE REACHES BREAKPOINT THIS BATTLE

JOB LIST

 WO  TD  V U H  FP   MD                        DAM.REC

217 6.37 1 4 2  .97  .14  .95 0.   0.   0.   0.   0.    .87  .09  .61  .66  .85

SIMULATION ENDED AFTER  6.4 DAYS

   HERE ARE THE RESULTS FOR RECOVERY AND EVACUATION

     191 VEHICLES RECOVERED  230 VEHICLES NEEDED RECOVERY

     MEAN NUMBER OF VEHICLES RECOVERED PER ATTACK 12.

     MEAN NUMBER OF VEHICLES NEEDING RECOVERY PER ATTACK  14.

   HERE ARE THE MAINTENANCE RESULTS

      NUMBER OF JOBS RECEIVED              217
```

NUMBER OF JOBS REPAIRED 65

NUMBER OF JOBS EVACUATED 50

NUMBER OF SUCCESSFUL CANNIBALIZATIONS 11

AVERAGE REPAIR CYCLE TIME 4.17 DAYS

AVERAGE REPAIR TIME FOR AUTOMOTIVE JOBS WAS 11.1524 HOURS

AVERAGE REPAIR TIME FOR ARMAMENT JOBS WAS 6.2216 HOURS

AVERAGE TIME A JOB WAITS FOR PARTS IS 11.1512 HOURS

AVERAGE TIME A JOB WAITS FOR INSPECTION IS 1.3914 HOURS (COMPANY)

MEASURES OF EFFECTIVENESS

RED CAS/BLUE CAS = 5.116

PERCENT OF RECOVERED JOBS REPAIRED .340

PERCENT OF DAMAGED VEHICLES REPAIRED .244

PERCENT OF TROOPS NOT KILLED .640

BACKLOG FOR MAIN

WAITING INSPECTION AT MAIN

WO NUM	TIME DOWN	VEH TYPE	UNIT	FP DAM	MOB DAM
155	0.	2	3	.9333	.6743

NO JOBS WAITING SHOP

WAITING TRANSPORTATION AT MAIN

WO NUM	TIME DOWN	VEH TYPE	UNIT	FP DAM	MOB DAM
147	0.	1	3	.9552	.1723
113	0.	2	1	.2804	.8141
124	0.	2	2	.6521	.8571
156	0.	2	4	.3818	.8778

WAITING PARTS AT MAIN

WO NUM	TIME DOWN	VEH TYPE	UNIT	FP DAM	MOB DAM
97	0.	2	2	.3941	.5915
135	0.	2	4	.1205	.6227

IN SHOP (ARMAMENT) AT MAIN

WO NUM	TIME DOWN	VEH TYPE	UNIT	FP DAM	MOB DAM	REP TIME
141	0.	1	2	.5940	.7479	8.2928
128	0.	2	3	.6461	.6837	6.9604
134	0.	2	2	.3075	.2450	1.3471

IN SHOP (AUTOMOTIVE) AT MAIN

WO NUM	TIME DOWN	VEH TYPE	UNIT	FP DAM	MOB DAM	REP TIME

139	0.	1	4	0.	.0304	20.5840
133	0.	1	4	0.	.4103	10.4295
141	0.	1	2	.5940	.7479	9.9444
128	0.	2	3	.6461	.6837	7.7269
134	0.	2	2	.3075	.2450	18.5374

BACKLOG FOR FWD1

WAITING INSPECTION AT FWD1

WO NUM	TIME DOWN	VEH TYPE	UNIT	FP DAM	MOB DAM
210	0.	1	1	.0308	.5506
211	0.	1	1	.2475	.6243
212	0.	1	1	.8192	.3491
189	0.	2	1	0.	.0767
202	0.	2	1	.7814	.8975

WAITING SHOP AT FWD1

WO NUM	TIME DOWN	VEH TYPE	UNIT	FP DAM	MOB DAM
52	0.	1	1	0.	.0309
110	0.	1	1	0.	.5487
119	0.	1	1	0.	.1866
143	0.	1	1	0.	.0623
76	0.	2	1	0.	.1479
95	0.	2	1	0.	.3865
185	0.	2	1	0.	.2819

WAITING TRANSPORTATION AT FWD1

WO NUM	TIME DOWN	VEH TYPE	UNIT	FP DAM	MOB DAM
188	0.	2	1	.2262	.9219
205	0.	1	1	.6000	.5082

NO JOBS WAITING PARTS
NO JOBS IN ARMAMENT SHOP
NO JOBS IN AUTOMOTIVE SHOP

BACKLOG FOR FWD2

NO JOBS WAITING INSPECTION

WAITING SHOP AT FWD2

WO NUM	TIME DOWN	VEH TYPE	UNIT	FP DAM	MOB DAM
111	0.	2	2	0.	.4023
140	0.	2	2	0.	.2325
151	0.	2	2	0.	.2636
157	0.	2	2	0.	.1006

WAITING TRANSPORTATION AT FWD2

WO NUM	TIME DOWN	VEH TYPE	UNIT	FP DAM	MOB DAM
195	0.	1	2	.5370	.7056
158	0.	2	2	.7794	.6800
196	0.	1	2	.4281	.0943
178	0.	2	2	.8010	.5495
190	0.	2	2	.1727	.2396
194	0.	2	2	.6126	.1180
206	0.	2	2	.3139	.7078
208	0.	2	2	.2111	.1736
213	0.	2	2	.4058	.0206

209	0.	1	2	.0856	.5803

NO JOBS WAITING PARTS

IN SHOP (ARMAMENT) AT FWD2

WO NUM	TIME DOWN	VEH TYPE	UNIT	FP DAM	MOB DAM	REP TIME
196	0.	1	2	.4281	.0943	6.0638

IN SHOP (AUTOMOTIVE) AT FWD2

WO NUM	TIME DOWN	VEH TYPE	UNIT	FP DAM	MOB DAM	REP TIME
102	0.	2	2	0.	.0164	5.0004

BACKLOG FOR FWD3

WAITING INSPECTION AT FWD3

WO NUM	TIME DOWN	VEH TYPE	UNIT	FP DAM	MOB DAM
171	0.	2	3	.0097	0.
176	0.	2	3	.8651	.4123
180	0.	2	3	.3657	.2482
186	0.	2	3	.0566	.6706
193	0.	2	3	.4657	.1126
199	0.	2	3	.7617	.2719
200	0.	2	3	.0711	.9776
214	0.	2	3	.8838	.3414

WAITING SHOP AT FWD3

WO NUM	TIME DOWN	VEH TYPE	UNIT	FP DAM	MOB DAM
120	0.	1	3	0.	.2829
131	0.	1	3	0.	.1759
137	0.	1	3	0.	.1273
138	0.	2	3	0.	.2794

WAITING TRANSPORTATION AT FWD3

WO NUM	TIME DOWN	VEH TYPE	UNIT	FP DAM	MOB DAM
184	0.	1	3	.8650	.9770
187	0.	1	3	.7067	.5995

NO JOBS WAITING PARTS
NO JOBS IN ARMAMENT SHOP
NO JOBS IN AUTOMOTIVE SHOP

BACKLOG FOR FWD4

WAITING INSPECTION AT FWD4

WO NUM	TIME DOWN	VEH TYPE	UNIT	FP DAM	MOB DAM
197	0.	2	4	.8364	.8107
198	0.	2	4	.9373	.0643
201	0.	2	4	.8595	.0864
203	0.	2	4	.7883	.4855
204	0.	2	4	.5272	.5263
215	0.	2	4	.5777	.0716

WAITING SHOP AT FWD4

WO NUM	TIME DOWN	VEH TYPE	UNIT	FP DAM	MOB DAM
100	0.	1	4	0.	.0769
154	0.	2	4	0.	.0726
160	0.	2	4	.0592	.4125

177	0.	2	4	0.	.1781
191	0.	2	4	.5135	.0152

WAITING TRANSPORTATION AT FWD4

WO NUM	TIME DOWN	VEH TYPE	UNIT	FP DAM	MOB DAM
29	0.	1	2	.8089	.1347

NO JOBS WAITING PARTS

IN SHOP (ARMAMENT) AT FWD4

WO NUM	TIME DOWN	VEH TYPE	UNIT	FP DAM	MOB DAM	REP TIME
160	0.	2	4	.0592	.4125	14.3494
191	0.	2	4	.5135	.0152	13.6226

NO JOBS IN AUTOMOTIVE SHOP

THE FOLLOWING JOBS HAVE BEEN COMPLETED:

WO NUM	TIME DOWN	VEH TYPE	UNIT	FP DAM	MOB DAM	AUTO REP	ARM REP	REP AT
1	0.	2	2	0.	0.	5.7248	0.	2
4	0.	1	2	0.	0.	0.	.1076	2
6	0.	2	4	0.	0.	0.	2.0532	4
5	0.	1	1	0.	0.	2.0655	0.	1
3	0.	2	3	0.	0.	2.2309	0.	0
2	0.	1	2	0.	0.	5.8866	0.	0
11	0.	1	1	0.	0.	6.1459	0.	1
14	0.	2	1	0.	0.	3.5945	0.	1
10	0.	2	4	0.	0.	0.	5.9131	0
13	0.	1	2	0.	0.	8.7578	0.	2
8	0.	1	3	0.	0.	12.2385	0.	0
9	0.	1	3	0.	0.	18.1724	0.	0
12	0.	1	1	0.	0.	25.7168	0.	1
16	0.	1	2	0.	0.	1.7414	0.	0
15	0.	2	3	0.	0.	14.1959	0.	3
20	0.	1	2	0.	0.	0.	.5083	2
7	0.	1	4	0.	0.	16.2171	0.	0
23	0.	2	1	0.	0.	0.	10.6978	1
18	0.	1	1	0.	0.	7.7572	0.	0
44	0.	1	3	0.	0.	2.3076	1.3133	3
22	0.	2	4	0.	0.	8.1576	0.	0
24	0.	1	1	0.	0.	0.	2.2762	0
25	0.	1	4	0.	0.	0.	15.4030	0
56	0.	1	2	0.	0.	12.1650	1.5410	2
36	0.	2	2	0.	0.	1.8757	0.	0
70	0.	1	3	0.	0.	2.7689	.5854	3
32	0.	2	1	0.	0.	9.5479	0.	0
39	0.	2	3	0.	0.	5.2876	2.9111	0
19	0.	1	2	0.	0.	30.0731	0.	0
35	0.	1	2	0.	0.	11.8296	3.4182	0
38	0.	2	4	0.	0.	3.2750	12.3661	0
33	0.	2	2	0.	0.	27.5618	.1214	2
57	0.	1	2	0.	0.	6.6653	0.	0
26	0.	1	2	0.	0.	19.0655	23.5869	0
47	0.	2	2	0.	0.	16.0082	2.7009	0
81	0.	1	1	0.	0.	2.1513	6.1968	0
63	0.	1	3	0.	0.	7.9967	0.	0
74	0.	1	2	0.	0.	6.3459	1.4516	0

105	0.	1	2	0.	0.	0.	10.7765	2
90	0.	1	1	0.	0.	.7657	2.1823	0
69	0.	2	4	0.	0.	14.2280	.1585	0
46	0.	2	4	0.	0.	7.4802	11.9251	0
98	0.	1	2	0.	0.	9.9802	18.5285	2
53	0.	2	1	0.	0.	8.7638	10.0348	0
89	0.	1	2	0.	0.	9.8415	5.1845	0
54	0.	2	2	0.	0.	22.4500	8.9745	0
62	0.	2	3	0.	0.	15.2604	.4628	0
59	0.	1	3	0.	0.	31.2713	1.8079	0
75	0.	2	1	0.	0.	9.6250	5.3459	0
79	0.	2	2	0.	0.	7.6828	0.	0
73	0.	1	2	0.	0.	20.6522	18.6879	0
87	0.	1	1	0.	0.	19.7512	5.0342	0
78	0.	2	1	0.	0.	10.1537	4.7294	0
92	0.	2	2	0.	0.	17.7570	0.	2
118	0.	1	3	0.	0.	10.5660	5.0148	0
122	0.	1	4	0.	0.	0.	4.5768	0
115	0.	2	3	0.	0.	9.8951	9.4051	0
109	0.	1	2	0.	0.	17.4302	5.1087	0
114	0.	2	3	0.	0.	17.1149	4.2149	0
150	0.	1	2	0.	0.	7.5832	4.5343	2
107	0.	2	2	0.	0.	2.9708	4.2067	0
91	0.	2	2	0.	0.	6.5209	2.1897	0
130	0.	1	2	0.	0.	8.2830	2.4650	0
83	0.	2	2	0.	0.	1.1371	9.1688	0
144	0.	2	4	0.	0.	3.0360	5.5893	0

THE FOLLOWING JOBS HAVE BEEN EVACUATED REAR:

WO NUM	TIME DOWN	VEH TYPE	UNIT	FP DAM	MOB DAM
43	0.	1	4	0.	.0090
29	0.	1	2	.8089	.1347
42	0.	1	4	.7645	.0444
40	0.	1	3	.7800	.1516
27	0.	1	1	.7927	.4651
37	0.	1	4	.9120	.4105
50	0.	1	1	.8045	.4076
48	0.	2	1	.1542	.7692
60	0.	1	3	.7865	.1612
65	0.	1	4	.8375	.0576
49	0.	2	2	.8555	.9219
58	0.	1	2	.4275	.8917
61	0.	2	3	.9048	.7533
51	0.	2	2	.7702	.4653
67	0.	1	4	.2058	.7758
68	0.	1	3	.8981	.8668
71	0.	1	3	.7147	.8105
72	0.	2	2	.9232	.0570
86	0.	1	2	.7668	.0408
66	0.	1	3	.3024	.9286
80	0.	1	1	.8846	.1846
101	0.	1	3	.8486	.5910
104	0.	1	1	.8797	.7481
85	0.	1	1	.6209	.8895

96	0.	1	2	.2593	.9791
99	0.	1	3	.7780	.2588
94	0.	1	2	.8034	.6535
55	0.	2	2	.5426	.7933
108	0.	2	3	.9230	.9291
77	0.	2	1	.7854	.3168
112	0.	1	3	.4152	.9686
103	0.	2	3	.5766	.7591
64	0.	2	3	.8883	.9905
116	0.	1	3	.8607	.7109
82	0.	2	1	.1882	.9388
84	0.	2	1	.6668	.9217
123	0.	1	4	.5048	.9210
93	0.	2	2	.9970	.1640
88	0.	2	1	.7312	.9880
136	0.	1	3	.8730	.9749
125	0.	1	4	.6937	.9923
121	0.	2	1	.1732	.8721
146	0.	1	3	.7504	.2891
129	0.	1	1	.5954	.8389
145	0.	1	1	.9249	.1654
117	0.	2	2	.9415	.0558
132	0.	2	1	.2285	.8189
106	0.	2	1	.9796	.9552
126	0.	2	1	.2192	.9131
142	0.	1	2	.7481	.8960

APPENDIX E PROGRAM LISTING

```
PREAMBLE
   NORMALLY MODE IS INTEGER
   THE SYSTEM OWNS A SUP.BN
     THE SYSTEM CAN OWN A CLOSED.JOB AND AN EVAC.JOB AND A KILL.JOB
   EVENT NOTICES INCLUDE FAILURE, BATTLE,DAYLIGHT AND STOP.SIMULATION
     EVERY BREAK HAS A SPEC.DAM AND A BN
     EVERY PARTS.COME HAS A SPEC.PART AND A LEV.PART
     EVERY ARRIVAL HAS A SPEC.ARR AND A LEV.ARR
     EVERY DIAGNOSIS HAS A SPEC.DIAG AND A LEV.DIAG
     EVERY MOVE.REAR HAS A SPEC.JOB AND A LEVEL
     EVERY REPAIR HAS AN A.CREW, A SPEC.REP AND A LEV.REP
     EVERY JUMP HAS A LEV.JUMP
     EVERY GET.THERE HAS A LEV.GET
   TEMPORARY ENTITIES.....
   EVERY MAINT.UNIT OWNS A SHOP, A WS.QUEUE, A WP.QUEUE, A WI.QUEUE, A WT.QUEUE,
     AN ARMAMENT, AN AUTOMOTIVE, HAS AN INSPECTOR,A NAME, A VEH.COUNT, A D.FLOT
     AND A NM.FOLKS, A NF.FOLKS, A T.JUMP AND BELONGS TO A SUP.BN
   EVERY JOB HAS A VEH.TYPE, A WO.NUM, A UNIT, A TIME.DOWN, A MOB.DAM,A LOOP.CN,
     A T.ARM.REP, A T.AUTO.REP, A REP.UNIT, AN IN.CAN, A TOT.DAM, A CAN.NUM,
     A FP.DAM, MAY BELONG TO A WS.QUEUE, A WP.QUEUE, A WI.QUEUE, A CLOSED.JOB,
     AN EVAC.JOB, AN ARMAMENT, AN AUTOMOTIVE, A WT.QUEUE, A KILL.JOB
   EVERY CREW HAS A MISSION, AN OCCUPATION, AN N.FOLKS AND BELONGS TO A SHOP
     DEFINE WI.QUEUE AS A SET RANKED BY LOW VEH.TYPE AND THEN
       BY LOW TIME.DOWN
     DEFINE WS.QUEUE AS A SET RANKED BY LOW VEH.TYPE AND THEN
       BY LOW TIME.DOWN
   DEFINE T.ARM.REP AND T.AUTO.REP AS REAL VARIABLES
   DEFINE T.PART.COMES AS A REAL VARIABLE
     DEFINE TIME.DOWN, MOB.DAM, FP.DAM AS REAL VARIABLES
   DEFINE X.RAT, R.VEHS, NMC.FOLKS, NFC.FOLKS, NFF.FOLKS, NMF.FOLKS AS REAL
     VARIABLES
   DEFINE N.FIGHT, REP.COUNT, TRY AS VARIABLES
   DEFINE COMP.TIMES AS A ROUTINE
   DEFINE CANNIBAL AS A ROUTINE GIVEN 2 ARGUMENT YIELDING 1 VALUE
   DEFINE SUBSTITUTE AS A ROUTINE GIVEN 3 ARGUMENTS
   DEFINE ASSESS.DAM AS A ROUTINE GIVEN 1 ARGUMENT
   DEFINE DET.ALLOC AS A ROUTINE
   DEFINE T.JUMP AND B.DIST AS REAL VARIABLES
   DEFINE CON.SPEED AND SETUP.TIME AS REAL VARIABLES
   DEFINE VEH.COUNT, V.CO.INIT, V.FS.INIT, D.FLOT, AND ALFA AS REAL VARIABLES
   DEFINE CAN.FIX AS A VARIABLE
   DEFINE A AND T.ACTION AS 2-DIMENSIONAL REAL ARRAYS
   DEFINE BIES, P.TANK, BATTLE.TIME, BUST, P.MOB, P.FIX.FWD,
     PR.HAVE.PARTS.FWD, PR.REAR.HAVE.PARTS, P.CO.FIX AS REAL
     VARIABLES
   DEFINE ARM.REP.TIME AND M.REP.TIME AS REAL VARIABLES
   TALLY MEAN.ARM.REP AS THE MEAN OF ARM.REP.TIME
   TALLY MEAN.AUTO.REP AS THE MEAN OF M.REP.TIME
   DEFINE TI.TIME AS A REAL VARIABLE
   TALLY MEAN.TI.TIME AS THE MEAN OF TI.TIME
```

```
DEFINE DOWN.TIME AS A REAL VARIABLE
DEFINE TOT.FOLKS AND CAS.COUNT AND R.CAS.CNT  AS REAL VARIABLES
DEFINE BAT.NUM AS A VARIABLE
TALLY S.CAS AS THE SUM OF CAS.COUNT
TALLY S.R.CAS AS THE SUM OF R.CAS.CNT
TALLY MEAN.DOWN.TIME AS THE MEAN OF DOWN.TIME
TALLY AVG.WP.TIME AS THE MEAN OF T.PART.COMES
DEFINE CO.MAINT TO MEAN 0
DEFINE DET1.MAINT TO MEAN 1
DEFINE DET2.MAINT TO MEAN 2
DEFINE DET3.MAINT TO MEAN 3
DEFINE DET4.MAINT TO MEAN 4
DEFINE IDLE TO MEAN 0
DEFINE BUSY TO MEAN 1
DEFINE DEAD TO MEAN 2
DEFINE TANK TO MEAN 1
DEFINE APC TO MEAN 2
DEFINE AUTO TO MEAN 1
DEFINE ARM TO MEAN 2
DEFINE SHOT TO MEAN 1
DEFINE SYS.FAIL TO MEAN 0
DEFINE NUM.EVAC.REAR AND  NUM.RET.BATTLE AS VARIABLES
DEFINE N.FOLKS, NM.FOLKS, NF.FOLKS AS REAL VARIABLES
DEFINE PK.PERS AND PK.TRUCK AS REAL VARIABLES
DEFINE S.ECH AS A REAL VARIABLE
DEFINE COUNT AS A VARIABLE
DEFINE DAM.REC AS A REAL 2-DIMENSIONAL ARRAY
DEFINE CAN.REC AS A 2-DIMENSIONAL ARRAY
DEFINE TOT.DAM AS A REAL VARIABLE
DEFINE COD AS A REAL VARIABLE
DEFINE EX.RAT, BZERO, B.ALIVE, RZERO, R.ALIVE, BP, R.2ECH, CCSL, CCSU,
  MCPO, SPACE.ECH, SELF.LIKE, UNREC, REC.NUM, TH, LS, LOS.PR, TGT.PRI,
  PR.INC.ID, LEAD.TIME, MTTF, USE.PER, ATT.CONST, FLOT.MOVE AS REAL VARIABLES
DEFINE DAY TO MEAN 1.0
DEFINE NIGHT TO MEAN 0.5
DEFINE D AS A REAL VARIABLE
DEFINE  LIGHT.STAT AS A REAL VARIABLE
DEFINE WORK.ORDER AND N.BNS AS VARIABLES
DEFINE TOT.REC AS A VARIABLE
DEFINE TOT.NEED.REC AS A VARIABLE
DEFINE B1.ALIVE AND B2.ALIVE AS REAL VARIABLES
DEFINE WHO.FIGHT AS A VARIABLE
DEFINE REC1.NUM AND REC2.NUM AS REAL VARIABLES
DEFINE MCPO1 AND MCPO2 AS REAL VARIABLES
DEFINE MCPO.ZERO AS A REAL VARIABLE
DEFINE QUIT1 AND QUIT2 AS VARIABLES
DEFINE QUIT AS A VARIABLE
DEFINE PR.DAY.INC AS A REAL VARIABLE
DEFINE SHOTF TO MEAN 2
TALLY SUM.REC AS THE SUM AND MEAN.REC AS THE MEAN OF TOT.REC
```

```
        TALLY SUM.NEED.REC AS THE SUM  AND AVG.NEED AS THE MEAN OF TOT.NEED.REC
        DEFINE SEED AS A VARIABLE
    END
```

```
MAIN
   RESERVE DAM.REC(*,*) AS 550 BY 11
   RESERVE CAN.REC(*,*) AS 100 BY 11
   RESERVE A(*,*) AS 8 BY 2
   RESERVE T.ACTION(*,*) AS 8 BY 3
   DEFINE NM AS A REAL VARIABLE
   DEFINE I,J,K,N.ARM,N.FWD.DET,N.GROUP AS VARIABLES
   DEFINE X AS A REAL VARIABLE
   DEFINE F, M, NF, LOF AND LOM AS REAL VARIABLES
   READ REP.COUNT
   READ SEED
   LET SEED.V(1)=SEED
   FOR I=2 TO 9, DO
     LET X=RANDOM.F(1)
     LET SEED.V(I)=SEED.V(1)+100
   LOOP
   LET X=RANDOM.F(1)
   LET SEED.V(1)=SEED.V(1)+100
   READ P.TANK
   READ W.FIGHT
   READ X.RAT, BZERO, AZERO, BP, A.2ECH, CCSL, CCSU
   READ MCPD.ZERO
   READ S.ECH, SELF.LIKE, UNREC, R.VEHS, TH
   READ LOS.PR, TGT.PRI, PR.INC.ID, LEAD.TIME
   READ MTTF, USE.PER
   READ D
   READ COD AND ALFA
   READ CON.SPEED AND SETUP.TIME  AND B.DIST
   READ PK.PERS
   READ NMC.FOLKS, NFC.FOLKS, V.CO.INIT, N.FWD.DET
   READ NMF.FOLKS, NFF.FOLKS, V.FS.INIT
   READ N.BNS, P.MOB, P.FIX.FWD, PR.HAVE.PARTS.FWD, PR.REAR.HAVE.PARTS,P.CO.FIX
   CALL COMP.TIMES
   LET LIGHT.STAT=DAY
   LET PR.DAY.INC=PR.INC.ID
   LET TOT.FOLKS=NMC.FOLKS +NFC.FOLKS+4.*(NMF.FOLKS+NFF.FOLKS)
   FOR TRY=1 TO REP.COUNT, DO
   LET BAT.NUM=0
   LET LS=LOS.PR
   LET WORK.ORDER=0
   LET EX.RAT=X.RAT
   LET WHO.FIGHT=W.FIGHT
   LET MCPD=MCPD.ZERO
   LET SPACE.ECH=S.ECH
   LET REC.NUM=R.VEHS
   LET NM=NMF.FOLKS
   LET NF=NFF.FOLKS
   IF LIGHT.STAT=NIGHT
     LET LIGHT.STAT=DAY
     LET CCSU=2.*CCSU
```

```
   LET CCSL=2.*CCSL
   LET SETUP.TIME=.5*SETUP.TIME
   LET PR.INC.ID=PR.DAY.INC
   LET CON.SPEED=2.*CON.SPEED
   LET TH=TH/1.5   ALWAYS
 IF TRY=1
 CALL INIT.PRINT
 ALWAYS
 LET ATT.CONST=-(1.0/BZERO)*LOG.E.F(EX.RAT)
 FOR K=1 TO BZERO*2. SCHEDULE A FAILURE IN EXPONENTIAL.F(MTTF/USE.PER,1)
   HOURS  ''COMPUTES FAILURE TIMES FOR ALL VEHICLES''
 LET R.ALIVE=RZERO
   LET B1.ALIVE=BZERO
   LET B2.ALIVE=BZERO
 LET REC1.NUM=REC.NUM/2.
 LET REC2.NUM=REC.NUM/2.
 LET MCPD1=MCPD.ZERO
 LET MCPD2=MCPD.ZERO
 IF WHO.FIGHT=3  LET B.ALIVE=2.*BZERO  ALWAYS
 CREATE A MAINT.UNIT  FILE MAINT.UNIT IN SUP.BN
   LET NAME(MAINT.UNIT)=CO.MAINT
   LET NM.FOLKS(MAINT.UNIT)=NMC.FOLKS
   LET NF.FOLKS(MAINT.UNIT)=NFC.FOLKS
   LET VEH.COUNT(MAINT.UNIT)=V.CO.INIT
   LET D.FLOT(MAINT.UNIT)=COD
    LET INSPECTOR(MAINT.UNIT)=2
 LET M=NM.FOLKS(MAINT.UNIT)/2.
 LET F=NF.FOLKS(MAINT.UNIT)/2.
 LET N.ARM=TRUNC.F(F)
 LET N.GROUP=TRUNC.F(M)+N.ARM
 IF FRAC.F(M)>0. LET LOM=1. ALWAYS
 IF FRAC.F(F)>0. LET LOF=1. ALWAYS
 FOR I=1 TO N.GROUP, DO
     CREATE A CREW   FILE CREW IN SHOP(MAINT.UNIT)
     IF I LE N.ARM  LET MISSION(CREW)=ARM
     LET N.FOLKS(CREW)=2.+LOF  LET LOF=0.
     ELSE LET MISSION(CREW)=AUTO
     LET N.FOLKS(CREW)=2.+LOM  LET LOM=0.
      ALWAYS LET OCCUPATION(CREW)=IDLE
 LOOP
 FOR I=1 TO N.FWD.DET, DO
 CREATE A MAINT.UNIT  FILE MAINT.UNIT IN SUP.BN
     LET NAME(MAINT.UNIT)=I
   LET NM.FOLKS(MAINT.UNIT)=NM
   LET NF.FOLKS(MAINT.UNIT)=NF
 LET M=NM.FOLKS(MAINT.UNIT)/2.
 LET F=NF.FOLKS(MAINT.UNIT)/2.
 LET N.ARM=TRUNC.F(F)
 LET N.GROUP=TRUNC.F(M)+N.ARM
 IF FRAC.F(M)>0. LET LOM=1. ALWAYS
```

```
      IF FRAC.F(F)>0. LET LOF=1. ALWAYS
        LET VEH.COUNT(MAINT.UNIT)=V.FS.INIT
        LET D.FLOT(MAINT.UNIT)=MCPD.ZERO
         LET INSPECTOR(MAINT.UNIT)=1
      FOR J=1 TO N.GROUP, DO
            CREATE A CREW   FILE CREW IN SHOP(MAINT.UNIT)
        IF J LE N.ARM LET MISSION(CREW)=ARM
            LET N.FOLKS(CREW)=2.+LOF   LET LOF=0.
           ELSE LET MISSION(CREW)=AUTO
          LET N.FOLKS(CREW)=2.+LOM  LET LOM=0.
           ALWAYS LET OCCUPATION(CREW)=IDLE
        LOOP LOOP
      SCHEDULE A BATTLE IN LEAD.TIME HOURS
      SCHEDULE A DAYLIGHT IN LEAD.TIME+ 15. HOURS
      PRINT 4 LINES AS FOLLOWS
   JOB LIST

    WO  TO  V U M  FP   MO                        DAM.REC

      START SIMULATION
      LOOP
      STOP
   END
```

```
EVENT FAILURE
   DEFINE SPEC.DAM AND BN AS VARIABLES
   DEFINE TEAM AS A VARIABLE
   IF TIME.V*HOURS.V LT LEAD.TIME
     LET SPEC.DAM=SYS.FAIL
     LET BN=RANDI.F(1,4,2)
     IF BN=1 OR BN=2 SUBTRACT 1 FROM B1.ALIVE  ALWAYS
     IF BN=3 OR BN=4 SUBTRACT 1 FROM B2.ALIVE  ALWAYS
     IF WHO.FIGHT=3    SUBTRACT 1 FROM B.ALIVE ALWAYS
     SCHEDULE A BREAK GIVEN SPEC.DAM AND BN IN UNIFORM.F(2.,3.,2) HOURS
     RETURN
   ELSE IF WHO.FIGHT=3
     IF RANDOM.F(2)>B.ALIVE/(BZERO*2.)  RETURN
     ELSE ADD 1 TO QUIT
       SUBTRACT 1. FROM B.ALIVE
       RETURN
     ELSE LET TEAM=RANDI.F(1,2,2)
     IF TEAM=1
       IF RANDOM.F(2)>B1.ALIVE/BZERO RETURN ELSE
       SUBTRACT 1. FROM B1.ALIVE
       ADD 1 TO QUIT1
       RETURN
     ELSE IF RANDOM.F(2)>B2.ALIVE/BZERO  RETURN  ELSE
       SUBTRACT 1. FROM B2.ALIVE
       ADD 1 TO QUIT2
     RETURN
END
```

```
EVENT BATTLE
   DEFINE REC.KILL AS A VARIABLE
   DEFINE I AS A VARIABLE
   DEFINE BROKE, DEST, DIFF, FLAG, K, KILLED, HOW.DAM, LEV.JUMP, NUM.JOBS,
     RELKS, TEMP AS VARIABLES
   DEFINE TE.R.TRANS AS A REAL VARIABLE
   DEFINE TB, RI, TRR, C, TEMP2, NR, T.REC, TE.TRANS, TE.HOOK, REC.TIME
     AS REAL VARIABLES
   DEFINE CHECK AS A REAL VARIABLE
   DEFINE BN, BNH, BNL AS VARIABLES
   DEFINE RAT AS A REAL VARIABLE
   ADD 1 TO BAT.NUM
   ADD 1 TO COUNT
   IF COUNT=4 LET COUNT=0 LET SPACE.ECH=20.  ALWAYS
   PRINT 3 LINES WITH BAT.NUM THUS

RESULTS OF BATTLE **

   CALL DET.ALLOC
   IF WHO.FIGHT=3
     LET BNL=1
     LET BNH=4
     GO AROUND
   ELSE IF WHO.FIGHT=1
     LET BNL=1
     LET BNH=2
     LET B.ALIVE=B1.ALIVE
     LET REC.NUM=REC1.NUM
     LET MCPD=MCPD1
     LET QUIT=QUIT1
   ELSE LET B.ALIVE=B2.ALIVE
     LET BNL=3
     LET BNH=4
     LET REC.NUM=REC2.NUM
     LET MCPD=MCPD2
     LET QUIT=QUIT2
   ALWAYS 'AROUND'
   LET EX.RAT=EXP.F(-ATT.CONST*B.ALIVE)*LIGHT.STAT
   LET RAT=R.ALIVE/B.ALIVE
   LET CHECK=EX.RAT-(((R.ALIVE/B.ALIVE)**2)*(1.-BP**2))
   IF CHECK LT 0.
     LET TB=-(1./SQRT.F(EX.RAT))*LOG.E.F(1.-BP)
   ELSE LET TB=LOG.E.F((SQRT.F(CHECK)-(RAT*BP))/(SQRT.F(EX.RAT)-RAT))/
     SQRT.F(EX.RAT)
   ALWAYS
   PRINT 3 LINES WITH WHO.FIGHT AND TB THUS

   WHO.FIGHT IS *   TIME OF BATTLE IS *.*** HOURS

   LET RI=R.2ECH*UNIFORM.F(0.5,1.0,2)
```

```
LET TRA=(SPACE.ECH/RI)+TB+(UNIFORM.F(5.,10.,2)/60.)
LET C=-EX.RAT*(TB/((1.-BP)*A.ALIVE))
LET REC.TIME=TB+TRA+C
LET TEMP2=((1.-BP)*A.ALIVE/EX.RAT)+REAL.F(QUIT)
LET CAS.COUNT=TEMP2-REAL.F(QUIT)
IF TEMP2 > B.ALIVE LET TEMP2=B.ALIVE+REAL.F(QUIT)   ALWAYS
LET NR=(1.-SELF.LIKE-UNREC)*TEMP2
IF REC.NUM=0 LET FLAG=1 GO DONE ALWAYS
LET T.REC=(NR/REC.NUM)*((MCPD*(1.+D)/CCSL)+TH+(MCPD*(1.+D)/CCSU))
IF REC.TIME LE T.REC
LET RECKS=INT.F(NR*REC.TIME/T.REC)
ELSE LET RECKS=INT.F(NR) ALWAYS
LET TE.TRANS=(MCPD/CCSU)*(1.+D)
LET TE.A.TRANS=(MCPD/CCSL)*(1.+D)
LET TE.HOOK=LS*UNIFORM.F(0.,.4,2)*(1./TGT.PRI)*PA.INC.ID*TH
LET NUM.JOBS=0
LET SPACE.ECH=S.ECH
IF RECKS GE REC.NUM
  LET TEMP=REC.NUM
  LET DIFF=RECKS-TEMP
ELSE LET TEMP=RECKS
ALWAYS FOR K=1 TO TEMP, DO
  IF RANDOM.F(2) LE (TAN.F(TE.TRANS*0.017453))
    OR RANDOM.F(2) LE ABS.F(1./LOG.E.F(TE.HOOK))
  OR RANDOM.F(2) LE TAN.F(TE.A.TRANS*0.017453)
    SUBTRACT 1 FROM REC.NUM
    ADD 1 TO REC.KILL
    IF REC.NUM=0.  GO DONE ALWAYS
    GO ON
  ELSE ADD 1 TO NUM.JOBS
'ON'  LOOP
IF DIFF NE 0
  FOR K=1 TO DIFF, DO
  IF RANDOM.F(2) LE (TAN.F(TE.TRANS*0.017453))
    OR RANDOM.F(2) LE ABS.F(1./LOG.E.F(TE.HOOK))
  OR RANDOM.F(2) LE TAN.F(TE.A.TRANS*0.017453)
    SUBTRACT 1 FROM REC.NUM
    ADD 1 TO REC.KILL
    IF REC.NUM=0.  GO DONE ALWAYS
    GO OUT
    ELSE ADD 1 TO NUM.JOBS
  'OUT'  LOOP
REGARDLESS  'DONE'
PRINT 2 LINES WITH REC.KILL THUS
NUMBER RECOVERY VEHICLES KILLED THIS BATTLE:  **

''SELF-LIKE, SYS FAIL''
LET BROKE=INT.F(SELF.LIKE*REAL.F(QUIT))
IF BROKE GE 1
  FOR I=1 TO BROKE, DO
```

```
        LET HOW.DAM=SYS.FAIL
        LET BN=RANDI.F(BNL,BNH,2)
        SCHEDULE A BREAK GIVEN HOW.DAM AND BN  IN UNIFORM.F(.5*TE.TRANS,
          REC.TIME,2) HOURS
      LOOP
    ELSE LET BROKE=0
    ALWAYS ''SELF-LIKE, SHOT''
    LET KILLED=INT.F(SELF.LIKE*(1.-BP)*R.ALIVE/EX.RAT)
    IF KILLED GE 1
      FOR I=1 TO KILLED, DO
        LET HOW.DAM=SHOTF
        LET BN=RANDI.F(BNL,BNH,2)
        SCHEDULE A BREAK GIVEN HOW.DAM AND BN  IN UNIFORM.F(.5*TE.TRANS,
          REC.TIME,2) HOURS
      LOOP
    ELSE LET KILLED=0
    ALWAYS  ''RECOVERED, SYS FAIL''
    IF FLAG=1 GO CHANGE ALWAYS
    LET BROKE=QUIT-BROKE
    IF BROKE GE 1
      FOR I=1 TO BROKE, DO
        LET HOW.DAM=SYS.FAIL
        LET BN=RANDI.F(BNL,BNH,2)
        SCHEDULE A BREAK GIVEN HOW.DAM AND BN  IN UNIFORM.F(.5*TE.TRANS,
          REC.TIME,2) HOURS
      LOOP
    ELSE LET BROKE=0
    ALWAYS  ''RECOVERED, SHOT''
    LET DEST=NUM.JOBS-BROKE
    FOR I=1 TO DEST, DO
      LET HOW.DAM=SHOT
        LET BN=RANDI.F(BNL,BNH,2)
        SCHEDULE A BREAK GIVEN HOW.DAM AND BN  IN UNIFORM.F(.5*TE.TRANS,
          REC.TIME,2) HOURS
    LOOP
    'CHANGE'
    LET B.ALIVE=B.ALIVE-TEMP2+REAL.F(QUIT)
    LET MCPD1=MCPD1-(FLOT.MOVE*TB)
    LET MCPD2=MCPD2-(FLOT.MOVE*TB)
    LET MCPD =MCPD -(FLOT.MOVE*TB)
    LET TOT.REC=NUM.JOBS
    LET TOT.NEED.REC=INT.F(NR)
    LET R.CAS.CNT=R.ALIVE*(1.-BP)
    LET R.ALIVE=BP*R.ALIVE
    IF R.ALIVE LE 70. ADD RZERO TO R.ALIVE
    ELSE ADD RZERO/2. TO R.ALIVE       ALWAYS
    LET SPACE.ECH=S.ECH
    IF WHO.FIGHT=3  GO AHEAD
    ELSE IF WHO.FIGHT=1
      SUBTRACT 4. FROM MCPD1
```

```
   LET QUIT1=0
   IF MCPD1 LE B.DIST
    ADD MCPD.ZERO TO MCPD1
    FOR I=1 TO 2, DO
     FOR EACH MAINT.UNIT IN SUP.BN WITH NAME(MAINT.UNIT)=I, DO
      LET LEV.JUMP=MAINT.UNIT
      SCHEDULE A JUMP GIVEN LEV.JUMP IN 15. MINUTES
      ADD MCPD.ZERO TO D.FLOT(MAINT.UNIT)
     LOOP
    LOOP ALWAYS
   LET REC1.NUM=REC.NUM
   LET B1.ALIVE=B.ALIVE
   GO AHEAD
  ELSE SUBTRACT 4. FROM MCPD2
   LET QUIT2=0
   IF MCPD2 LE B.DIST
    ADD MCPD.ZERO TO MCPD2
    FOR I=3 TO 4, DO
     FOR EACH MAINT.UNIT IN SUP.BN WITH NAME(MAINT.UNIT)=I, DO
      LET LEV.JUMP=MAINT.UNIT
      SCHEDULE A JUMP GIVEN LEV.JUMP IN 15. MINUTES
      ADD MCPD.ZERO TO D.FLOT(MAINT.UNIT)
     LOOP
    LOOP ALWAYS
   LET REC2.NUM=REC.NUM
   LET B2.ALIVE=B.ALIVE
  'AHEAD'
  FOR EACH MAINT.UNIT IN SUP.BN WITH NAME(MAINT.UNIT)>0, DO
   IF NAME(MAINT.UNIT) LE 2 LET D.FLOT(MAINT.UNIT)=MCPD1
   ELSE LET D.FLOT(MAINT.UNIT)=MCPD2
  ALWAYS LOOP
  FOR EACH MAINT.UNIT IN SUP.BN WITH NAME(MAINT.UNIT)=CO.MAINT
   LET D.FLOT(MAINT.UNIT)=D.FLOT(MAINT.UNIT)-(FLOT.MOVE*TB)
  PRINT 2 LINES WITH REC.NUM, B.ALIVE, R.ALIVE THUS
  * RE. VEHS. LEFT: **  B.ALIVE: ***  R.ALIVE: ***

  IF B.ALIVE LE 71.
   IF WHO.FIGHT NE 3
    LET WHO.FIGHT=3
    LET B.ALIVE=B1.ALIVE+B2.ALIVE
    LET QUIT=QUIT1+QUIT2
    LET REC.NUM=REC1.NUM+REC2.NUM
   SCHEDULE A BATTLE IN TB+TRR+(4.-TB)/RI HOURS
   GO WRITE
   ELSE SCHEDULE A STOP.SIMULATION IN TB HOURS
    PRINT 1 LINE THUS
BLUE REACHES BREAKPOINT THIS BATTLE
   GO WRITE
  ELSE
   IF WHO.FIGHT=1  LET WHO.FIGHT=2 ELSE
```

```
        IF WHO.FIGHT=2  LET WHO.FIGHT=1  ALWAYS  REGARDLESS
        SCHEDULE A BATTLE IN TB+TRA+(4.-TB)/RI HOURS
      LET QUIT=0
      'WRITE'
      PRINT 5 LINES AS FOLLOWS

   JOB LIST

    NO  TD  V U H  FP   MD                         DAM.REC

      RETURN
   END
```

```
EVENT BREAK GIVEN HOW.DAM AND BN
   DEFINE LEVEL AS A VARIABLE
   DEFINE BN, WO AS VARIABLES
   DEFINE SPEC.JOB AND HOW.DAM AS VARIABLES
   DEFINE X AS A REAL VARIABLE
   DEFINE MAX.DAM AS A REAL VARIABLE
   LET X=RANDOM.F(2)
   LET WORK.ORDER=WORK.ORDER+1
   CREATE A JOB
     LET UNIT(JOB)=BN
     LET WO.NUM(JOB)=WORK.ORDER
     IF X LE P.TANK LET VEH.TYPE(JOB)=TANK
     ELSE LET VEH.TYPE(JOB)=APC
   ALWAYS
     IF HOW.DAM=SYS.FAIL  GO ON
     ELSE IF HOW.DAM=SHOT
       LET MAX.DAM=1.0
     ELSE LET MAX.DAM=0.2   ALWAYS
     IF VEH.TYPE(JOB)=TANK
       LET FP.DAM(JOB)=UNIFORM.F(0.,1.,2)
       LET MOB.DAM(JOB)=UNIFORM.F(0.,MAX.DAM,3)
     GO DOWN
     ELSE LET FP.DAM(JOB)=UNIFORM.F(0., 1., 2)
       LET MOB.DAM(JOB)=UNIFORM.F(0.,MAX.DAM,3)
     GO DOWN
   'ON'  LET X=RANDOM.F(3)
       IF X LE P.MOB LET MOB.DAM(JOB)=UNIFORM.F(0.,.2,3)
       ELSE LET FP.DAM(JOB)=UNIFORM.F(0.,.2,2)
       ALWAYS
   'DOWN'
   LET SPEC.JOB=JOB
   CALL ASSESS.DAM GIVEN SPEC.JOB
   LET WO=WO.NUM(JOB)
   IF VEH.TYPE(JOB)=TANK
   PRINT 1 LINE WITH WORK.ORDER, TIME.V, VEH.TYPE(JOB), UNIT(JOB), HOW.DAM,
     FP.DAM(JOB), MOB.DAM(JOB), DAM.REC(WO,1),DAM.REC(WO,2), DAM.REC(WO,3),
     DAM.REC(WO,4),DAM.REC(WO,5),DAM.REC(WO,6),DAM.REC(WO,7),DAM.REC(WO,8),
     DAM.REC(WO,9),DAM.REC(WO,10),DAM.REC(WO,11)      THUS
*** *.** * * * *.** *.** *.** *.** *.** *.** *.** *.** *.** *.** *.** *.** *.**
   ELSE
   PRINT 1 LINE WITH WORK.ORDER, TIME.V, VEH.TYPE(JOB), UNIT(JOB), HOW.DAM,
     FP.DAM(JOB), MOB.DAM(JOB), DAM.REC(WO,1),DAM.REC(WO,2), DAM.REC(WO,3),
     DAM.REC(WO,4),DAM.REC(WO,5),DAM.REC(WO,6),DAM.REC(WO,7),DAM.REC(WO,8),
     DAM.REC(WO,9)   THUS
*** *.** * * * *.** *.** *.** *.** *.** *.** *.** *.** *.** *.** *.**
   ALWAYS
   FOR EACH MAINT.UNIT IN SUP.BN, DO
       IF NAME(MAINT.UNIT)=UNIT(JOB)  GO AHEAD
       ELSE  LOOP
   'AHEAD'  LET SPEC.JOB=JOB  LET LEVEL=MAINT.UNIT
```

```
    IF T.JUMP(MAINT.UNIT) NE 0.0  AND TIME.V LT T.JUMP(MAINT.UNIT)
      SCHEDULE AN ARRIVAL GIVEN SPEC.JOB AND LEVEL AT T.JUMP(MAINT.UNIT)
    ELSE SCHEDULE AN ARRIVAL GIVEN SPEC.JOB AND LEVEL NOW
    ALWAYS RETURN
END
```

```
EVENT ARRIVAL GIVEN SPEC.JOB AND LEVEL
   DEFINE LEV.MOVE  AS A VARIABLE
   DEFINE SPEC.JOB AND LEVEL AS VARIABLES
   LET JOB=SPEC.JOB  LET MAINT.UNIT=LEVEL
   ADD 1. TO VEH.COUNT(MAINT.UNIT)
   IF (NM.FOLKS(MAINT.UNIT) LE 1. AND MOB.DAM(JOB) NE 0.) OR
     (NF.FOLKS(MAINT.UNIT) LE 1. AND FP.DAM(JOB) NE 0.)
     LET LEV.MOVE=MAINT.UNIT
     SCHEDULE A MOVE.REAR GIVEN SPEC.JOB AND LEV.MOVE IN (BETA.F(A(7,1),
       A(7,2),9)*(T.ACTION(7,3)-T.ACTION(7,1)))+T.ACTION(7,1) HOURS
     FILE JOB IN WT.QUEUE(MAINT.UNIT)  RETURN  ELSE
   IF INSPECTOR(MAINT.UNIT)=0
      FILE JOB IN WI.QUEUE(MAINT.UNIT)  RETURN
   ELSE SUBTRACT 1 FROM INSPECTOR(MAINT.UNIT)
   IF NAME(MAINT.UNIT)>0
       LET TI.TIME=(BETA.F(A(1,1),A(1,2),4)*(T.ACTION(1,3)-T.ACTION(1,1)))+
       T.ACTION(1,1)
       SCHEDULE A DIAGNOSIS GIVEN SPEC.JOB AND LEVEL IN TI.TIME HOURS
      RETURN
   ''AT COMPANY''
     ELSE LET TI.TIME=(BETA.F(A(2,1),A(2,2),4)*(T.ACTION(2,3)-T.ACTION(2,1)))+
       T.ACTION(2,1)
     SCHEDULE A DIAGNOSIS GIVEN SPEC.JOB AND LEVEL IN TI.TIME HOURS
   RETURN
END
```

```
EVENT REPAIR GIVEN A.CREW, SPEC.JOB AND LEVEL
   DEFINE WO AS A VARIABLE
   DEFINE JOB.CAN AS A VARIABLE
   DEFINE CAN AS A VARIABLE
   DEFINE SPEC.JOB AND LEVEL AS VARIABLES
   DEFINE A.CREW AS A VARIABLE
   LET CREW=A.CREW  LET JOB=SPEC.JOB
   LET WO = WO.NUM(JOB)
   LET MAINT.UNIT=LEVEL
   IF IN.CAN(JOB) NE 0
     CALL SUBSTITUTE GIVEN SPEC.JOB, LEVEL AND A.CREW
     ADD 1 TO CAN.FIX
   ALWAYS
   IF MISSION(CREW)=AUTO  LET MOB.DAM(JOB)=0.
     IF JOB IS IN AUTOMOTIVE REMOVE JOB FROM AUTOMOTIVE  ALWAYS
   ELSE LET FP.DAM(JOB)=0.
     IF JOB IS IN ARMAMENT REMOVE JOB FROM ARMAMENT    ALWAYS
   ALWAYS  IF FP.DAM(JOB)=0. AND MOB.DAM(JOB)=0.
     IF JOB IS IN CLOSED.JOB GO LOOK OTHERWISE
     LET DOWN.TIME=TIME.V-TIME.DOWN(JOB)
     ADD 1 TO NUM.RET.BATTLE  FILE JOB IN CLOSED.JOB
   IF JOB IS IN AUTOMOTIVE REMOVE JOB FROM AUTOMOTIVE ALWAYS
   IF JOB IS IN ARMAMENT   REMOVE JOB FROM ARMAMENT   ALWAYS
   IF JOB IS IN WT.QUEUE   REMOVE JOB FROM WT.QUEUE   ALWAYS
   IF JOB IS IN WS.QUEUE   REMOVE JOB FROM WS.QUEUE   ALWAYS
   IF JOB IS IN WI.QUEUE   REMOVE JOB FROM WI.QUEUE   ALWAYS
   IF JOB IS IN WP.QUEUE   REMOVE JOB FROM WP.QUEUE   ALWAYS
     SUBTRACT 1. FROM VEH.COUNT(MAINT.UNIT)
       IF WHO.FIGHT=3 ADD 1 TO B.ALIVE  GO LOOK
       ELSE IF UNIT(JOB)=1 OR UNIT(JOB)=2  ADD 1 TO B1.ALIVE  GO LOOK
       ELSE ADD 1 TO B2.ALIVE  GO LOOK
       ELSE IF JOB IS NOT IN WS.QUEUE AND JOB IS NOT IN AUTOMOTIVE
         AND JOB IS NOT IN ARMAMENT
     FILE JOB IN WS.QUEUE(MAINT.UNIT)
     ELSE 'LOOK' IF WS.QUEUE(MAINT.UNIT) IS EMPTY
       GO CONTROL
     ELSE FOR EACH JOB IN WS.QUEUE(MAINT.UNIT), DO
       IF MISSION(CREW)=AUTO AND MOB.DAM(JOB)>0.
         AND JOB NOT IN AUTOMOTIVE  GO TAKE
       ELSE IF MISSION(CREW)=ARM AND FP.DAM(JOB)>0.
         AND JOB NOT IN ARMAMENT  GO TAKE
       REGARDLESS  ALWAYS LOOP
   'CONTROL' ''TRY TO CANNIBALIZE''
   IF WP.QUEUE IS EMPTY AND WT.QUEUE IS EMPTY
     LET OCCUPATION(CREW)=IDLE  RETURN
   ELSE FOR EACH JOB IN WP.QUEUE(MAINT.UNIT), DO
     IF (FP.DAM(JOB)=0. AND MISSION(CREW)=ARM) OR (MOB.DAM(JOB)=0. AND
         MISSION(CREW)=ARM)  GO DOWN
     ELSE
     LET JOB.CAN=JOB
```

```
          CALL CANNIBAL GIVEN JOB.CAN AND LEVEL YIELDING CAN
           LET SPEC.JOB=JOB
           LET MAINT.UNIT=LEVEL
            IF CAN=0  GO DOWN
          ELSE
        CALL SUBSTITUTE GIVEN SPEC.JOB, LEVEL AND A.CREW
          IF MISSION(CREW)=ARM
          LET T.ARM.REP=(BETA.F(A(4,1),A(4,2),6)*(T.ACTION(4,3)-T.ACTION(4,1)))+
            T.ACTION(4,1)
          SCHEDULE A REPAIR GIVEN A.CREW, SPEC.JOB AND LEVEL IN T.ARM.REP HOURS
          FILE JOB IN ARMAMENT(MAINT.UNIT)
          ELSE
          LET T.AUTO.REP=(BETA.F(A(3,1),A(3,2),6)*(T.ACTION(3,3)-T.ACTION(3,1)))+
              T.ACTION(3,1)
          SCHEDULE A REPAIR GIVEN A.CREW, SPEC.JOB AND LEVEL IN T.AUTO.REP HOURS
          FILE JOB IN AUTOMOTIVE(MAINT.UNIT)
        ALWAYS RETURN
        'DOWN'  LOOP
        IF NAME(MAINT.UNIT)=CO.MAINT  LET OCCUPATION(CREW)=IDLE RETURN
        ELSE FOR EACH JOB IN WT.QUEUE(MAINT.UNIT), DO
          IF JOB IS IN AUTOMOTIVE OR JOB IS IN ARMAMENT
            FOR EACH MOVE.REAR IN EV.S(I.MOVE.REAR) WITH WO.NUM(SPEC.MOVE) EQ
              WO.NUM(JOB), DO
              CANCEL THE MOVE.REAR DESTROY THE MOVE.REAR
              REMOVE THE JOB FROM WT.QUEUE(MAINT.UNIT) LOOP GO OUT ALWAYS
          IF (FP.DAM(JOB)=0. AND MISSION(CREW)=ARM) OR (MOB.DAM(JOB)=0. AND
            MISSION(CREW)=AUTO)  GO OUT
          ELSE
          LET JOB.CAN=JOB
              CALL CANNIBAL GIVEN JOB.CAN AND LEVEL YIELDING CAN
           LET SPEC.JOB=JOB
           LET MAINT.UNIT=LEVEL
           IF CAN=0  GO OUT
          ELSE
          CALL SUBSTITUTE GIVEN SPEC.JOB, LEVEL AND A.CREW
          IF MISSION(CREW)=ARM
          LET T.ARM.REP=(BETA.F(A(4,1),A(4,2),6)*(T.ACTION(4,3)-T.ACTION(4,1)))+
            T.ACTION(4,1)
          SCHEDULE A REPAIR GIVEN A.CREW, SPEC.JOB AND LEVEL IN T.ARM.REP HOURS
          FILE JOB IN ARMAMENT(MAINT.UNIT)
          ELSE
          LET T.AUTO.REP=(BETA.F(A(3,1),A(3,2),6)*(T.ACTION(3,3)-T.ACTION(3,1)))+
              T.ACTION(3,1)
          SCHEDULE A REPAIR GIVEN A.CREW, SPEC.JOB AND LEVEL IN T.AUTO.REP HOURS
          FILE JOB IN AUTOMOTIVE(MAINT.UNIT)
        ALWAYS RETURN
        'OUT'  LOOP
        LET OCCUPATION(CREW)=IDLE  RETURN
        'TAKE'  REMOVE JOB FROM WS.QUEUE(MAINT.UNIT)
            LET SPEC.JOB=JOB
```

```
    IF MISSION(CREW)=ARM AND FP.DAM(JOB) NE 0.
      LET T.ARM.REP=(BETA.F(A(4,1),A(4,2),6)*(T.ACTION(4,3)-T.ACTION(4,1)))+
        T.ACTION(4,1)
      LET REP.UNIT(JOB)=NAME(MAINT.UNIT)
      SCHEDULE A REPAIR GIVEN A.CREW, SPEC.JOB, LEVEL IN T.ARM.REP(JOB) HOURS
      LET ARM.REP.TIME=T.ARM.REP(JOB)
        FILE JOB IN ARMAMENT(MAINT.UNIT)   RETURN
    ELSE IF MISSION(CREW)=AUTO AND MOB.DAM(JOB) NE 0.
      LET T.AUTO.REP=(BETA.F(A(3,1),A(3,2),6)*(T.ACTION(3,3)-T.ACTION(3,1)))+
          T.ACTION(3,1)
      LET REP.UNIT(JOB)=NAME(MAINT.UNIT)
      SCHEDULE A REPAIR GIVEN A.CREW, SPEC.JOB, LEVEL IN T.AUTO.REP(JOB) HOURS
      LET M.REP.TIME=T.AUTO.REP(JOB)
         FILE JOB IN AUTOMOTIVE(MAINT.UNIT)    RETURN
    ELSE RETURN
END
```

```
EVENT PARTS.COME GIVEN SPEC.JOB AND LEVEL
   DEFINE BAY AND FBAY AS VARIABLES
   DEFINE SPEC.JOB AND LEVEL AS VARIABLES
   DEFINE A.CREW AS A VARIABLE
   LET MAINT.UNIT=LEVEL  LET JOB=SPEC.JOB
   IF JOB IS IN WS.QUEUE RETURN ELSE
   IF JOB IS NOT IN WP.QUEUE  RETURN   ELSE
   REMOVE THIS JOB FROM WP.QUEUE(MAINT.UNIT)
   IF JOB IS IN AUTOMOTIVE GO DOWN  OTHERWISE
   IF MOB.DAM(JOB)>0.
     FOR EACH CREW IN SHOP(MAINT.UNIT) WITH MISSION(CREW)=AUTO,DO
       IF OCCUPATION(CREW)=IDLE  LET BAY=1  LET A.CREW=CREW
       GO OUTSIDE
       ELSE  LOOP
   'OUTSIDE'  IF BAY=1
     LET T.AUTO.REP=(BETA.F(A(3,1),A(3,2),6)*(T.ACTION(3,3)-T.ACTION(3,1)))+
         T.ACTION(3,1)
     LET REP.UNIT(JOB)=NAME(MAINT.UNIT)
     SCHEDULE A REPAIR GIVEN A.CREW, SPEC.JOB, LEVEL IN T.AUTO.REP(JOB) HOURS
     LET M.REP.TIME=T.AUTO.REP(JOB)
     FILE JOB IN AUTOMOTIVE(MAINT.UNIT)
   ELSE FILE JOB IN WS.QUEUE(MAINT.UNIT)
   REGARDLESS ALWAYS 'DOWN' IF JOB IS IN ARMAMENT GO ON  OTHERWISE
   IF FP.DAM(JOB)=0. GO BEYOND ELSE
     FOR EACH CREW IN SHOP(MAINT.UNIT) WITH MISSION(CREW)=ARM, DO
       IF OCCUPATION(CREW)=IDLE LET FBAY=1 LET A.CREW=CREW
       GO BEYOND
       ELSE  LOOP
   'BEYOND'  IF FBAY=1
     LET T.ARM.REP=(BETA.F(A(4,1),A(4,2),6)*(T.ACTION(4,3)-T.ACTION(4,1)))+
       T.ACTION(4,1)
     LET REP.UNIT(JOB)=NAME(MAINT.UNIT)
     SCHEDULE A REPAIR GIVEN A.CREW, SPEC.JOB, LEVEL IN T.ARM.REP(JOB) HOURS
     LET ARM.REP.TIME=T.ARM.REP(JOB)
     FILE JOB IN ARMAMENT(MAINT.UNIT)
   ELSE IF JOB IS NOT IN WS.QUEUE
         FILE JOB IN WS.QUEUE(MAINT.UNIT)
   REGARDLESS ALWAYS 'ON' RETURN
END
```

```
EVENT DIAGNOSIS GIVEN SPEC.JOB AND LEV.DIAG
   DEFINE LEV.ARR AS A VARIABLE
   DEFINE LEVEL AS A VARIABLE
   DEFINE BAY, FBAY, CAN, FP, LEV.DIAG, SPEC.JOB AS VARIABLES
   DEFINE A.CREW AS A VARIABLE
   DEFINE P.PARTS, X AND P.FIX AS REAL VARIABLES
   DEFINE CAN.VEH AS A 1-DIMENSIONAL ARRAY
   LET LEVEL =LEV.DIAG
   LET JOB=SPEC.JOB  LET MAINT.UNIT=LEVEL
   IF VEH.TYPE(JOB)=TANK  LET FP=11
   ELSE LET FP=9
   ALWAYS ADD 1 TO INSPECTOR(MAINT.UNIT)
   IF NAME(MAINT.UNIT)>0 LET P.FIX=P.FIX.FWD
     LET P.PARTS=PR.HAVE.PARTS.FWD
   ELSE LET P.FIX=P.CO.FIX  LET P.PARTS=PR.REAR.HAVE.PARTS
   ALWAYS IF MOB.DAM(JOB)>P.FIX OR FP.DAM(JOB)>P.FIX GO EVAC.MAYBE
   ELSE LET X=UNIFORM.F(0., 1., 4)
   IF X>P.PARTS ''NEED PARTS, TRY TO CANNIBALIZE''
     CALL CANNIBAL GIVEN SPEC.JOB AND LEVEL YIELDING CAN
      LET MAINT.UNIT=LEVEL
      LET JOB=SPEC.JOB
     IF CAN=0  GO EVAC.MAYBE
     OTHERWISE ELSE ''HAVE PARTS'' IF MOB.DAM(JOB)>0.0
       FOR EACH CREW IN SHOP(MAINT.UNIT) WITH MISSION(CREW)=AUTO, DO
         IF OCCUPATION(CREW)=IDLE  LET BAY=1 LET A.CREW=CREW
            GO FIX
         ELSE  LOOP
   'FIX'  IF BAY=1
     LET T.AUTO.REP=(BETA.F(A(3,1),A(3,2),6)*(T.ACTION(3,3)-T.ACTION(3,1)))+
         T.ACTION(3,1)
     LET REP.UNIT(JOB)=NAME(MAINT.UNIT)
     SCHEDULE A REPAIR GIVEN A.CREW, SPEC.JOB, LEVEL IN T.AUTO.REP(JOB) HOURS
     LET M.REP.TIME=T.AUTO.REP(JOB)
          FILE JOB IN AUTOMOTIVE(MAINT.UNIT)
            ELSE FILE JOB IN WS.QUEUE(MAINT.UNIT)
   REGARDLESS
   ALWAYS REGARDLESS  IF FP.DAM(JOB)=0.   GO NEXT
   ELSE     FOR EACH CREW IN SHOP(MAINT.UNIT) WITH MISSION(CREW)=ARM, DO
              IF OCCUPATION(CREW)=IDLE LET FBAY=1 LET A.CREW=CREW
                GO BEYOND
              ELSE  LOOP
   'BEYOND'  IF FBAY=1
     LET T.ARM.REP=(BETA.F(A(4,1),A(4,2),6)*(T.ACTION(4,3)-T.ACTION(4,1)))+
       T.ACTION(4,1)
     LET REP.UNIT(JOB)=NAME(MAINT.UNIT)
     SCHEDULE A REPAIR GIVEN A.CREW, SPEC.JOB, LEVEL IN T.ARM.REP(JOB) HOURS
     LET ARM.REP.TIME=T.ARM.REP(JOB)
         FILE JOB IN ARMAMENT(MAINT.UNIT)    GO NEXT
         ELSE IF JOB IS NOT IN WS.QUEUE
          FILE JOB IN WS.QUEUE(MAINT.UNIT)  GO NEXT
```

```
    'EVAC.MAYBE' REGARDLESS IF NAME(MAINT.UNIT)>0
       IF MOB.DAM(JOB)<.2 SCHEDULE A MOVE.REAR GIVEN SPEC.JOB AND LEVEL NOW
        ELSE SCHEDULE A MOVE.REAR GIVEN SPEC.JOB AND LEVEL IN
        (BETA.F(A(7,1),A(7,2),9)*(T.ACTION(7,3)-T.ACTION(7,1)))+T.ACTION(7,1)
        HOURS
      ALWAYS
       FILE JOB IN WT.QUEUE(MAINT.UNIT)
      GO NEXT
      ELSE IF MOB.DAM(JOB)>P.FIX OR FP.DAM(JOB)>P.FIX
        SCHEDULE A MOVE.REAR GIVEN SPEC.JOB AND LEVEL IN
        (BETA.F(A(8,1),A(8,2),9)*(T.ACTION(8,3)-T.ACTION(8,1)))+T.ACTION(8,1)
        HOURS
          FILE JOB IN WT.QUEUE(MAINT.UNIT)  GO NEXT
        ELSE FILE JOB IN WP.QUEUE(MAINT.UNIT)
          LET T.PART.COMES=(BETA.F(A(5,1),A(5,2),6)*(T.ACTION(5,3)-
          T.ACTION(5,1)))+T.ACTION(5,1)
          SCHEDULE A PARTS.COME GIVEN SPEC.JOB AND LEVEL IN
           T.PART.COMES HOURS
    'NEXT'
    LET LEVEL =LEV.DIAG
    IF WI.QUEUE IS EMPTY  RETURN
    ELSE REMOVE FIRST JOB FROM WI.QUEUE(LEVEL)  LET SPEC.JOB=JOB
    SUBTRACT 1 FROM INSPECTOR(LEVEL)
    IF NAME(LEVEL)>0
        LET TI.TIME=(BETA.F(A(1,1),A(1,2),4)*(T.ACTION(1,3)-T.ACTION(1,1)))+
        T.ACTION(1,1)
        SCHEDULE A DIAGNOSIS GIVEN SPEC.JOB AND LEVEL IN TI.TIME HOURS
       RETURN
    ''AT COMPANY''
      ELSE LET TI.TIME=(BETA.F(A(2,1),A(2,2),4)*(T.ACTION(2,3)-T.ACTION(2,1)))+
        T.ACTION(2,1)
      SCHEDULE A DIAGNOSIS GIVEN SPEC.JOB AND LEVEL IN TI.TIME HOURS  RETURN
 END
```

```
EVENT MOVE.REAR GIVEN SPEC.JOB AND LEVEL
   DEFINE LEV.ARR AS A VARIABLE
   DEFINE SPEC.JOB AND LEVEL AS VARIABLES
   LET MAINT.UNIT=LEVEL  LET JOB=SPEC.JOB
   IF THIS JOB IS NOT IN WT.QUEUE  RETURN
   ELSE SUBTRACT 1. FROM VEH.COUNT(MAINT.UNIT)
     REMOVE THIS JOB FROM WT.QUEUE(MAINT.UNIT)
   IF JOB IS IN AUTOMOTIVE REMOVE JOB FROM AUTOMOTIVE ALWAYS
   IF JOB IS IN ARMAMENT   REMOVE JOB FROM ARMAMENT   ALWAYS
   IF JOB IS IN WT.QUEUE   REMOVE JOB FROM WT.QUEUE   ALWAYS
   IF JOB IS IN WS.QUEUE   REMOVE JOB FROM WS.QUEUE   ALWAYS
   IF JOB IS IN WI.QUEUE   REMOVE JOB FROM WI.QUEUE   ALWAYS
   IF JOB IS IN WP.QUEUE   REMOVE JOB FROM WP.QUEUE   ALWAYS
     IF NAME(MAINT.UNIT)>CO.MAINT
     FOR EACH MAINT.UNIT IN SUP.BN, DO
       IF NAME(MAINT.UNIT)=CO.MAINT GO AHEAD
       ELSE LOOP
     'AHEAD' LET LEV.ARR=MAINT.UNIT
     LET MAINT.UNIT=LEVEL
     SCHEDULE AN ARRIVAL GIVEN SPEC.JOB AND LEV.ARR IN (BETA.F(A(6,1),A(6,2),9)
         *(T.ACTION(6,3)-T.ACTION(6,1)))+T.ACTION(6,1) HOURS
   ELSE
     IF JOB IS NOT IN EVAC.JOB FILE JOB IN EVAC.JOB
     ADD 1 TO NUM.EVAC.REAR ALWAYS
   REGARDLESS RETURN
END
```

```
EVENT DAYLIGHT
   IF TIME.V*HOURS.V LT LEAD.TIME  RETURN
   ELSE IF LIGHT.STAT = DAY
     LET LIGHT.STAT=NIGHT
     LET CCSL=.5*CCSL
     LET CCSU=.5*CCSU
     LET LS=.1
     LET TH=1.5*TH
     LET PR.INC.ID=.3
     LET D=2.*D
     LET SETUP.TIME=2.*SETUP.TIME
     LET CON.SPEED=.5*CON.SPEED
     SCHEDULE A DAYLIGHT IN 9 HOURS
     RETURN
   ELSE LET LIGHT.STAT=DAY
     LET CCSL=2.*CCSL
     LET CCSU=2.*CCSU
     LET LS=LOS.PR
     LET TH=TH/1.5
     LET PR.INC.ID=PR.DAY.INC
     LET D=0.5*D
     LET SETUP.TIME=.5*SETUP.TIME
     LET CON.SPEED=2.*CON.SPEED
     SCHEDULE A DAYLIGHT IN 15. HOURS
     RETURN
END
```

```
EVENT JUMP GIVEN LEV.JUMP
   DEFINE LEV.JUMP AS A VARIABLE
   DEFINE F.TIME AND J.TIME AS REAL VARIABLES
   LET MAINT.UNIT=LEV.JUMP
   LET J.TIME=MCPD.ZERO/CON.SPEED+SETUP.TIME/MINUTES.V
   SCHEDULE A GET.THERE GIVEN LEV.JUMP IN J.TIME HOURS
   FOR EACH JOB IN WS.QUEUE(MAINT.UNIT), DO
     IF MOB.DAM(JOB) GT 0.2 AND JOB IS NOT IN ARMAMENT AND JOB IS NOT IN
       AUTOMOTIVE
       REMOVE THE JOB FROM WS.QUEUE(MAINT.UNIT)
       DESTROY THE JOB
       SUBTRACT 1. FROM VEH.COUNT(MAINT.UNIT)
   ALWAYS LOOP
   FOR EACH JOB IN WI.QUEUE(MAINT.UNIT), DO
     IF MOB.DAM(JOB) GT 0.2
       REMOVE THE JOB FROM WI.QUEUE(MAINT.UNIT)
       DESTROY THE JOB
       SUBTRACT 1. FROM VEH.COUNT(MAINT.UNIT)
   ALWAYS LOOP
   FOR EACH JOB IN WT.QUEUE(MAINT.UNIT), DO
     FOR EACH MOVE.REAR IN EV.S(I.MOVE.REAR) WITH SPEC.JOB=JOB, DO
       IF TIME.A(MOVE.REAR)>(15./(HOURS.V*MINUTES.V))+TIME.V
         CANCEL THE MOVE.REAR  DESTROY THE MOVE.REAR
       IF JOB IS IN WS.QUEUE REMOVE JOB FROM WS.QUEUE ALWAYS
       IF JOB IS IN ARMAMENT REMOVE JOB FROM ARMAMENT  ALWAYS
       IF JOB IS IN AUTOMOTIVE REMOVE JOB FROM AUTOMOTIVE ALWAYS
       IF JOB IS IN WP.QUEUE REMOVE JOB FROM WP.QUEUE ALWAYS
       IF JOB IS IN WT.QUEUE REMOVE JOB FROM WT.QUEUE ALWAYS
         DESTROY THE JOB
         SUBTRACT 1. FROM VEH.COUNT(MAINT.UNIT)
     ALWAYS LOOP
   LOOP
   FOR EACH JOB IN ARMAMENT(MAINT.UNIT), DO
     FOR EACH REPAIR IN EV.S(I.REPAIR) WITH SPEC.REP(REPAIR)=JOB AND
       LEV.REP(REPAIR)=MAINT.UNIT, DO
       IF MOB.DAM(JOB)>0.0
         LET OCCUPATION(A.CREW)=IDLE
         CANCEL THE REPAIR  DESTROY THE REPAIR
         IF JOB IS IN ARMAMENT REMOVE JOB FROM ARMAMENT ALWAYS
         IF JOB IS IN WS.QUEUE REMOVE JOB FROM WS.QUEUE ALWAYS
         IF JOB IS IN AUTOMOTIVE REMOVE JOB FROM AUTOMOTIVE ALWAYS
         DESTROY THE JOB
         SUBTRACT 1. FROM VEH.COUNT(MAINT.UNIT)
       ELSE LET F.TIME = TIME.A(REPAIR)-TIME.V
       CANCEL THE REPAIR
         RESCHEDULE THIS REPAIR IN (F.TIME*HOURS.V)+J.TIME HOURS
     ALWAYS LOOP
   LOOP
   FOR EACH JOB IN AUTOMOTIVE(MAINT.UNIT), DO
     FOR EACH REPAIR IN EV.S(I.REPAIR) WITH SPEC.REP(REPAIR)=JOB AND
```

```
            LEV.REP(REPAIR)=MAINT.UNIT, DO
            LET OCCUPATION(A.CREW)=IDLE
            CANCEL THE REPAIR  DESTROY THE REPAIR
            REMOVE THE JOB FROM AUTOMOTIVE(MAINT.UNIT)
              IF JOB IS IN WS.QUEUE REMOVE JOB FROM WS.QUEUE ALWAYS
              IF JOB IS IN ARMAMENT REMOVE JOB FROM ARMAMENT ALWAYS
            DESTROY THE JOB
            SUBTRACT 1. FROM VEH.COUNT(MAINT.UNIT)
        LOOP LOOP
        LET T.JUMP(MAINT.UNIT)=TIME.V+J.TIME/HOURS.V
        RETURN
    END
```

```
EVENT GET.THERE GIVEN LEV.GET
   DEFINE LEV.GET, SPEC.JOB, CAN, LEV.MOVE, FP, A.CREW, J AS VARIABLES
   LET MAINT.UNIT=LEV.GET
   LET T.JUMP(MAINT.UNIT)=0.0
   FOR EACH JOB IN WS.QUEUE(MAINT.UNIT) WITH IN.CAN(JOB) NE 0, DO
     IF VEH.TYPE=TANK LET FP=11
     ELSE LET FP=9  ALWAYS
     FOR J=1 TO FP LET CAN.REC(IN.CAN(JOB),J)=0
     LET SPEC.JOB=JOB
     CALL CANNIBAL GIVEN SPEC.JOB AND LEV.GET YIELDING CAN
     IF CAN=0 REMOVE JOB FROM WS.QUEUE(MAINT.UNIT)
     FOR EACH MAINT.UNIT IN SUP.BN WITH NAME(MAINT.UNIT)=0
       LET LEV.MOVE=MAINT.UNIT
     IF MOB.DAM(JOB)<0.2 SCHEDULE A MOVE.REAR GIVEN SPEC.JOB AND
       LEV.MOVE NOW
       LET MAINT.UNIT=LEV.GET
       FILE JOB IN WT.QUEUE(MAINT.UNIT)
     ELSE SCHEDULE A MOVE.REAR GIVEN SPEC.JOB AND LEV.MOVE IN
       (BETA.F(A(7,1),A(7,2),9)*(T.ACTION(7,3)-T.ACTION(7,1)))
       +T.ACTION(7,1) HOURS
       FILE JOB IN WT.QUEUE(MAINT.UNIT)
     ALWAYS LET MAINT.UNIT=LEV.GET
     ALWAYS LOOP
   IF WS.QUEUE IS EMPTY GO ON ELSE
   FOR EACH CREW IN SHOP(MAINT.UNIT) WITH OCCUPATION(CREW)=IDLE, DO
     IF MISSION(CREW)=AUTO
       FOR EACH JOB IN WS.QUEUE(MAINT.UNIT), DO
         IF MOB.DAM(JOB)>0 AND JOB IS NOT IN ARMAMENT
           AND JOB IS NOT IN AUTOMOTIVE
           REMOVE JOB FROM WS.QUEUE
           FILE JOB IN AUTOMOTIVE  LET SPEC.JOB=JOB
           LET A.CREW=CREW
           LET T.AUTO.REP=(BETA.F(A(3,1),A(3,2),6)*(T.ACTION(3,3)-
             T.ACTION(3,1)))+T.ACTION(3,1)
           SCHEDULE A REPAIR GIVEN A.CREW, SPEC.JOB AND LEV.GET IN
             T.AUTO.REP HOURS
       ALWAYS LOOP
     ELSE FOR EACH JOB IN WS.QUEUE(MAINT.UNIT), DO
       IF FP.DAM(JOB)>0.0 AND JOB IS NOT IN AUTOMOTIVE
         AND JOB IS NOT IN ARMAMENT
         REMOVE JOB FROM WS.QUEUE(MAINT.UNIT)
         FILE JOB IN ARMAMENT(MAINT.UNIT)
         LET SPEC.JOB=JOB  LET A.CREW=CREW
         LET T.ARM.REP=(BETA.F(A(4,1),A(4,2),6)*(T.ACTION(4,3)-
           T.ACTION(4,1)))+T.ACTION(4,1)
         SCHEDULE A REPAIR GIVEN A.CREW, SPEC.JOB AND LEV.GET IN
           T.ARM.REP HOURS
       ALWAYS LOOP
     ALWAYS LOOP
   'ON' RETURN  END
```

```
EVENT STOP.SIMULATION
   DEFINE IDENT AS AN ALPHA VARIABLE
   DEFINE LOS.RAT, PER.FIX.REC, PR.REP.DAM, P.TROOPS, TROOPS AS REAL VARIABLES
   FOR EACH MAINT.UNIT IN SUP.BN, DO
     ADD NM.FOLKS(MAINT.UNIT) TO TROOPS
     ADD NF.FOLKS(MAINT.UNIT) TO TROOPS  LOOP
   LET LOS.RAT=S.R.CAS/S.CAS
   LET PER.FIX.REC=NUM.RET.BATTLE/SUM.REC
   LET PR.REP.DAM=NUM.RET.BATTLE/S.CAS
   LET P.TROOPS=TROOPS/TOT.FOLKS
   PRINT 5 LINES WITH TIME.V THUS

SIMULATION ENDED AFTER **.* DAYS

   HERE ARE THE RESULTS FOR RECOVERY AND EVACUATION

   PRINT 5 LINES WITH SUM.REC, SUM.NEED.REC, MEAN.REC AND AVG.NEED THUS
     *** VEHICLES RECOVERED  *** VEHICLES NEEDED RECOVERY

     MEAN NUMBER OF VEHICLES RECOVERED PER ATTACK **.

     MEAN NUMBER OF VEHICLES NEEDING RECOVERY PER ATTACK ***.
   PRINT 3 LINES THUS

   HERE ARE THE MAINTENANCE RESULTS

   PRINT 8 LINES WITH WORK.ORDER, NUM.RET.BATTLE, NUM.EVAC.REAR AND
     CAN.FIX THUS
      NUMBER OF JOBS RECEIVED            *****

      NUMBER OF JOBS REPAIRED            *****

      NUMBER OF JOBS EVACUATED           *****

      NUMBER OF SUCCESSFUL CANNIBALIZATIONS  ****

     PRINT 2 LINES WITH MEAN.DOWN.TIME THUS
      AVERAGE REPAIR CYCLE TIME          ***.** DAYS

   PRINT 4 LINES WITH MEAN.AUTO.REP AND MEAN.ARM.REP AS FOLLOWS
      AVERAGE REPAIR TIME FOR AUTOMOTIVE JOBS WAS **.**** HOURS

      AVERAGE REPAIR TIME FOR ARMAMENT JOBS WAS **.**** HOURS

   PRINT 4 LINES WITH AVG.WP.TIME AND MEAN.TI.TIME THUS
      AVERAGE TIME A JOB WAITS FOR PARTS IS **.**** HOURS

      AVERAGE TIME A JOB WAITS FOR INSPECTION IS **.**** HOURS (COMPANY)

   PRINT 10 LINES WITH LOS.RAT, PER.FIX.REC, PR.REP.DAM, P.TROOPS THUS
```

```
      MEASURES OF EFFECTIVENESS

        RED CAS/BLUE CAS = ***.***

        PERCENT OF RECOVERED JOBS REPAIRED  *.***

        PERCENT OF DAMAGED VEHICLES REPAIRED  *.***

        PERCENT OF TROOPS NOT KILLED  *.***

      FOR EACH MAINT.UNIT IN SUP.BN, DO
        IF NAME(MAINT.UNIT)=0  LET IDENT="MAINT.COMP"
        ALWAYS IF NAME(MAINT.UNIT)=1  LET IDENT="FWD1"
        ALWAYS IF NAME(MAINT.UNIT)=2  LET IDENT="FWD2"
        ALWAYS IF NAME(MAINT.UNIT)=3  LET IDENT="FWD3"
        ALWAYS IF NAME(MAINT.UNIT)=4  LET IDENT="FWD4"
        REGARDLESS
          PRINT 2 LINES WITH IDENT THUS
   BACKLOG FOR **********

        IF WI.QUEUE IS NOT EMPTY
          PRINT 2 LINES WITH IDENT THUS
              WAITING INSPECTION AT **********

           PRINT 1 LINE THUS
   WO NUM   TIME DOWN   VEH TYPE   UNIT      FP DAM   MOB DAM
            FOR EACH JOB IN WI.QUEUE, DO
          PRINT 1 LINE WITH WO.NUM, TIME.DOWN,VEH.TYPE, UNIT,FP.DAM, MOB.DAM THUS
    ****    ***.*****      *        *        *.****   *.****
        LOOP
        ELSE PRINT 1 LINE THUS
             NO JOBS WAITING INSPECTION
        ALWAYS IF WS.QUEUE IS NOT EMPTY
          PRINT 2 LINES WITH IDENT THUS
              WAITING SHOP AT **********

           PRINT 1 LINE THUS
   WO NUM   TIME DOWN   VEH TYPE   UNIT      FP DAM   MOB DAM
          FOR EACH JOB IN WS.QUEUE, DO
          PRINT 1 LINE WITH WO.NUM, TIME.DOWN,VEH.TYPE, UNIT,FP.DAM, MOB.DAM THUS
    ****    ***.*****      *        *        *.****   *.****
        LOOP
        ELSE PRINT 1 LINE THUS
             NO JOBS WAITING SHOP
          ALWAYS IF WT.QUEUE IS NOT EMPTY
          PRINT 2 LINES WITH IDENT THUS
              WAITING TRANSPORTATION AT **********

          PRINT 1 LINE THUS
   WO NUM   TIME DOWN   VEH TYPE   UNIT      FP DAM   MOB DAM
```

```
          FOR EACH JOB IN WT.QUEUE, DO
          PRINT 1 LINE WITH WO.NUM, TIME.DOWN,VEH.TYPE, UNIT,FP.DAM, MOB.DAM THUS
  ****    ***.*****      *       *       *.****   *.****
          LOOP
          ELSE PRINT 1 LINE AS FOLLOWS
             NO JOBS WAITING TRANSPORTATION
        ALWAYS IF WP.QUEUE IS NOT EMPTY
          PRINT 2 LINES WITH IDENT THUS
              WAITING PARTS AT **********

           PRINT 1 LINE THUS
 WO NUM   TIME DOWN   VEH TYPE   UNIT      FP DAM   MOB DAM
          FOR EACH JOB IN WP.QUEUE, DO
          PRINT 1 LINE WITH WO.NUM, TIME.DOWN,VEH.TYPE, UNIT,FP.DAM, MOB.DAM THUS
  ****    ***.*****      *       *       *.****   *.****
        LOOP
        ELSE PRINT 1 LINE THUS
             NO JOBS WAITING PARTS
        ALWAYS IF ARMAMENT IS NOT EMPTY
          PRINT 2 LINES WITH IDENT THUS

              IN SHOP (ARMAMENT) AT **********
           PRINT 1 LINE THUS
 WO NUM   TIME DOWN   VEH TYPE   UNIT      FP DAM   MOB DAM   REP TIME
        FOR EACH JOB IN ARMAMENT(MAINT.UNIT), DO
          PRINT 1 LINE WITH WO.NUM, TIME.DOWN,VEH.TYPE, UNIT,FP.DAM, MOB.DAM,
        T.ARM.REP  THUS
  ****    ***.*****      *       *       *.****   *.****    **.****
        LOOP
        ELSE PRINT 1 LINE THUS
             NO JOBS IN ARMAMENT SHOP
        ALWAYS IF AUTOMOTIVE IS NOT EMPTY
          PRINT 2 LINES WITH IDENT THUS

              IN SHOP (AUTOMOTIVE) AT **********
           PRINT 1 LINE THUS
 WO NUM   TIME DOWN   VEH TYPE   UNIT      FP DAM   MOB DAM   REP TIME
        FOR EACH JOB IN AUTOMOTIVE(MAINT.UNIT), DO
          PRINT 1 LINE WITH WO.NUM, TIME.DOWN,VEH.TYPE, UNIT,FP.DAM, MOB.DAM,
          T.AUTO.REP  THUS
  ****    ***.*****      *       *       *.****   *.****    **.****
        LOOP
          ELSE PRINT 1 LINE THUS
             NO JOBS IN AUTOMOTIVE SHOP
        ALWAYS LOOP
        PRINT 3 LINES THUS

 THE FOLLOWING JOBS HAVE BEEN COMPLETED:

        PRINT 1 LINE THUS
```

```
WO NUM  TIME DOWN  VEH TYPE  UNIT  FP DAM  MOB DAM  AUTO REP  ARM REP  REP AT
     FOR EACH JOB IN CLOSED.JOB, DO
       PRINT 1 LINE WITH WO.NUM, TIME.DOWN,VEH.TYPE, UNIT,FP.DAM, MOB.DAM,
        T.AUTO.REP, T.ARM.REP, REP.UNIT THUS
 ****   ***.****      *      *    *.****   *.****  **.****   **.****     *
     LOOP
     PRINT 3 LINES THUS

THE FOLLOWING JOBS HAVE BEEN EVACUATED REAR:

     PRINT 1 LINE THUS
WO NUM   TIME DOWN   VEH TYPE   UNIT      FP DAM   MOB DAM
     FOR EACH JOB IN EVAC.JOB, DO
       PRINT 1 LINE WITH WO.NUM, TIME.DOWN,VEH.TYPE, UNIT,FP.DAM, MOB.DAM THUS
 ****    ***.*****      *        *        *.****   *.****
     LOOP
   CALL START.OVER
   RETURN
END
```

```
ROUTINE START.OVER
   LET NUM.EVAC.REAR=0
   LET NUM.RET.BATTLE=0
   LET COUNT=0
   LET CAN.FIX=0
   LET TIME.V=0.0
   RESET TOTALS OF TOT.REC AND TOT.NEED.REC
   RESET TOTALS OF DOWN.TIME, M.REP.TIME, ARM.REP.TIME, TI.TIME
   RESET TOTALS OF CAS.COUNT AND R.CAS.CNT
   FOR EACH REPAIR IN EV.S(I.REPAIR), DO
     CANCEL THE REPAIR  DESTROY THE REPAIR  LOOP
   FOR EACH DAYLIGHT IN EV.S(I.DAYLIGHT), DO
     CANCEL THE DAYLIGHT  DESTROY THE DAYLIGHT LOOP
   FOR EACH MOVE.REAR IN EV.S(I.MOVE.REAR), DO
     CANCEL THE MOVE.REAR  DESTROY THE MOVE.REAR  LOOP
   FOR EACH DIAGNOSIS IN EV.S(I.DIAGNOSIS), DO
     FILE SPEC.DIAG(DIAGNOSIS) IN KILL.JOB
     CANCEL THE DIAGNOSIS  DESTROY THE DIAGNOSIS  LOOP
   FOR EACH JUMP IN EV.S(I.JUMP), DO
     CANCEL THE JUMP  DESTROY THE JUMP  LOOP
   FOR EACH GET.THERE IN EV.S(I.GET.THERE), DO
     CANCEL THE GET.THERE  DESTROY THE GET.THERE  LOOP
   FOR EACH ARRIVAL IN EV.S(I.ARRIVAL), DO
     CANCEL THE ARRIVAL  DESTROY THE ARRIVAL  LOOP
   FOR EACH BREAK IN EV.S(I.BREAK), DO
     CANCEL THE BREAK  DESTROY THE BREAK  LOOP
   FOR EACH FAILURE IN EV.S(I.FAILURE), DO
     CANCEL THE FAILURE  DESTROY THE FAILURE  LOOP
     FOR EACH BATTLE IN EV.S(I.BATTLE), DO
     CANCEL THE BATTLE  DESTROY THE BATTLE  LOOP
   FOR EACH PARTS.COME IN EV.S(I.PARTS.COME), DO
     CANCEL THE PARTS.COME  DESTROY THE PARTS.COME  LOOP
   FOR EACH MAINT.UNIT IN SUP.BN, DO
     FOR EACH JOB IN WT.QUEUE(MAINT.UNIT), DO
       IF JOB IS IN WT.QUEUE
       REMOVE THE JOB FROM WT.QUEUE(MAINT.UNIT)
       ALWAYS
       IF JOB IS NOT IN KILL.JOB FILE JOB IN KILL.JOB ALWAYS
     LOOP
   FOR EACH JOB IN WS.QUEUE(MAINT.UNIT), DO
       IF JOB IS IN WS.QUEUE
       REMOVE JOB FROM WS.QUEUE(MAINT.UNIT)
       ALWAYS
       IF JOB IS NOT IN KILL.JOB FILE JOB IN KILL.JOB ALWAYS
     LOOP
     FOR EACH JOB IN ARMAMENT(MAINT.UNIT), DO
       IF JOB IS IN ARMAMENT
       REMOVE JOB FROM ARMAMENT(MAINT.UNIT)
       ALWAYS
       IF JOB IS NOT IN KILL.JOB FILE JOB IN KILL.JOB ALWAYS
```

```
        LOOP
        FOR EACH JOB IN AUTOMOTIVE(MAINT.UNIT), DO
          IF JOB IS IN AUTOMOTIVE
          REMOVE JOB FROM AUTOMOTIVE(MAINT.UNIT)
          ALWAYS
          IF JOB IS NOT IN KILL.JOB FILE JOB IN KILL.JOB ALWAYS
        LOOP
        FOR EACH JOB IN WI.QUEUE(MAINT.UNIT), DO
          IF JOB IS IN WI.QUEUE
          REMOVE JOB FROM WI.QUEUE(MAINT.UNIT)
          ALWAYS
          IF JOB IS NOT IN KILL.JOB FILE JOB IN KILL.JOB ALWAYS
        LOOP
        FOR EACH JOB IN WP.QUEUE(MAINT.UNIT), DO
          IF JOB IS IN WP.QUEUE
          REMOVE JOB FROM WP.QUEUE(MAINT.UNIT)
          ALWAYS
          IF JOB IS NOT IN KILL.JOB FILE JOB IN KILL.JOB ALWAYS
        LOOP
      LOOP
      FOR EACH JOB IN CLOSED.JOB, DO
        REMOVE JOB FROM CLOSED.JOB
          IF JOB IS NOT IN KILL.JOB FILE JOB IN KILL.JOB ALWAYS
        LOOP
      FOR EACH JOB IN EVAC.JOB, DO
        REMOVE THE JOB FROM EVAC.JOB
        IF JOB IS NOT IN KILL.JOB FILE JOB IN KILL.JOB ALWAYS
      LOOP
        FOR EACH JOB IN KILL.JOB, DO
        REMOVE THE JOB FROM KILL.JOB  DESTROY THE JOB  LOOP
      FOR I=1 TO 100, FOR J=1 TO 11, LET CAN.REC(I,J)=0
      FOR I=1 TO 550, FOR J=1 TO 11, LET DAM.REC(I,J)=0.0
      FOR EACH MAINT.UNIT IN SUP.BN, DO
        REMOVE THE MAINT.UNIT FROM SUP.BN
        DESTROY THE MAINT.UNIT  LOOP
      RETURN
   END
```

```
ROUTINE TO ASSESS.DAM GIVEN SPEC.JOB
   DEFINE SPEC.JOB, WO, YCOUNT AS VARIABLES
   DEFINE MOB, FP, Z AND Y AS VARIABLES
   LET JOB=SPEC.JOB
   IF VEH.TYPE(JOB)=TANK  LET MOB=6  LET FP=11
   ELSE LET MOB=6  LET FP=9
   ALWAYS LET WO=WO.NUM(JOB)
   IF MOB.DAM(JOB) NE 0.
     LET Y=INT.F(MOB.DAM(JOB)*REAL.F(MOB))
       'DRAW'  LET Z=RANDI.F(1,MOB,8)
     IF DAM.REC(WO,Z) NE 0. GO DRAW
     ELSE LET DAM.REC(WO,Z)=UNIFORM.F(0.,1., 7)
     ADD 1 TO YCOUNT  IF YCOUNT LT Y GO DRAW
   REGARDLESS ALWAYS IF FP.DAM(JOB)NE 0.
     LET Y=INT.F(FP.DAM(JOB)*REAL.F(FP-MOB))
     LET YCOUNT=0
     'ROLL'  LET Z=RANDI.F((MOB+1),FP,8)
     IF DAM.REC(WO,Z) NE 0.  GO ROLL
     ELSE LET DAM.REC(WO,Z)=RANDOM.F(7)
       ADD 1 TO YCOUNT  IF YCOUNT LT Y GO ROLL
   REGARDLESS ALWAYS
   RETURN
END
```

```
ROUTINE CANNIBAL GIVEN SPEC.JOB AND LEVEL YIELDING CAN
   DEFINE FB, I, J, NSB, WO AS VARIABLES
   DEFINE SPEC.JOB AS A VARIABLE
   DEFINE Z AS A REAL VARIABLE
   DEFINE X AS A REAL VARIABLE
   DEFINE JUNK AS A VARIABLE
   DEFINE CAN.VEH AS A 1-DIMENSIONAL ARRAY
   DEFINE LEVEL AS A VARIABLE
   DEFINE CAN AS A VARIABLE
   LET JOB=SPEC.JOB  LET MAINT.UNIT=LEVEL
   LET WO=WO.NUM(JOB)
   IF VEH.TYPE(JOB)=TANK  LET FP=11
   ELSE LET FP=9
   ALWAYS RESERVE CAN.VEH(*) AS FP
   FOR I=1 TO FP, DO
     IF DAM.REC(WO,I)>0.  ADD 1 TO NSB
     ALWAYS LOOP
   FOR EACH JUNK IN WT.QUEUE WITH VEH.TYPE(JUNK)=VEH.TYPE(SPEC.JOB), DO
     IF JUNK=JOB  GO CONTINUE  ALWAYS
     FOR J=1 TO FP, DO
       IF DAM.REC(WO,J)=0. GO LOOP
       ELSE IF CAN.VEH(J) NE 0  GO LOOP
       ELSE IF DAM.REC(WO,J)>1.-DAM.REC(WO.NUM(JUNK),J)  GO LOOP
       ELSE LET X=RANDOM.F(1)
       IF X>1.0-DAM.REC(WO.NUM(JUNK),J)   GO LOOP
       ELSE LET CAN.VEH(J)=JUNK  ADD 1 TO CAN
       ADD 1 TO CAN.NUM(JUNK)  ADD 1.0 TO DAM.REC(WO.NUM(JUNK),J)
     'LOOP' LOOP
     IF CAN=NSB   GO AHEAD
   ELSE 'CONTINUE'  LOOP
   FOR EACH JUNK IN WP.QUEUE WITH VEH.TYPE(JUNK)=VEH.TYPE(SPEC.JOB), DO
     IF JUNK=JOB  GO ON  ALWAYS
     FOR J=1 TO FP, DO
       IF DAM.REC(WO,J)=0. GO AROUND
       ELSE IF CAN.VEH(J) NE 0 GO AROUND
       ELSE IF DAM.REC(WO,J)>1.-DAM.REC(WO.NUM(JUNK),J)  GO AROUND
       ELSE LET X=RANDOM.F(1)
       IF X>1.0-DAM.REC(WO.NUM(JUNK),J)   GO AROUND
       ELSE LET CAN.VEH(J)=JUNK  ADD 1 TO CAN
       ADD 1 TO CAN.NUM(JUNK)  ADD 1.0 TO DAM.REC(WO.NUM(JUNK),J)
     'AROUND' LOOP
     IF CAN=NSB   GO AHEAD
   ELSE 'ON' LOOP
   ''CANNOT CANNIBALIZE'' LET CAN=0
   RELEASE CAN.VEH(*)      RETURN
   'AHEAD'  ''PREPARE TO CANNIBALIZE''
     IF JOB IS IN WT.QUEUE   REMOVE JOB FROM WT.QUEUE     ALWAYS
       IF JOB IS IN WP.QUEUE
       REMOVE JOB FROM WP.QUEUE     ALWAYS
   IF JOB IS NOT IN WS.QUEUE FILE JOB IN WS.QUEUE ALWAYS
```

```
        FOR J=1 TO FP, DO
          IF CAN.VEH(J)=0 GO THRU
          ELSE FOR EACH MOVE.REAR IN EV.S(I.MOVE.REAR) WITH SPEC.JOB=CAN.VEH(J), DO
            CANCEL THIS MOVE.REAR  DESTROY THIS MOVE.REAR
        'THRU'  LOOP
          FOR EACH PARTS.COME IN EV.S(I.PARTS.COME) WITH SPEC.JOB=CAN.VEH(J), DO
             CANCEL THIS PARTS.COME
          LET Z=(BETA.F(A(5,1),A(5,2),6)*(T.ACTION(5,3)-T.ACTION(5,1)))+
            T.ACTION(5,1)
          LET Z=Z/24.0
            LET T.PART.COMES=MAX.F(TIME.A(PARTS.COME),TIME.V+Z) -TIME.V
            RESCHEDULE THIS PARTS.COME GIVEN CAN.VEH(J) AND LEVEL IN T.PART.COMES
             HOURS
          IF CAN.VEH(J) IS NOT IN WP.QUEUE FILE CAN.VEH(J) IN WP.QUEUE(MAINT.UNIT)
            ALWAYS
          LOOP
        LOOP
      FOR I=1 TO 100, DO
          FOR J=1 TO 11, DO
            IF CAN.REC(I,J) NE 0 GO DROP
          ELSE LOOP
          LET IN.CAN(JOB)=I  GO FURTHER
        'DROP' LOOP
        'FURTHER' FOR J=1 TO FP LET CAN.REC(IN.CAN(JOB),J)=CAN.VEH(J)
      RELEASE CAN.VEH(*)
      RETURN
   END
```

```
ROUTINE TO SUBSTITUTE GIVEN SPEC.JOB, LEVEL AND A.CREW
   DEFINE A.CREW, FP, J, LEVEL, M, N, WO AS VARIABLES
   DEFINE JOB.MOVE AS A VARIABLE
   LET MAINT.UNIT=LEVEL  LET JOB=SPEC.JOB  LET CREW=A.CREW
   LET WO=WO.NUM(JOB)
   IF VEH.TYPE(JOB)=TANK  LET FP=11
   ELSE LET FP=9
   ALWAYS IF MISSION(CREW)=AUTO  LET M=1  LET N=6
   ELSE LET M=7  LET N=FP
   ALWAYS FOR J=M TO N, DO
     IF CAN.REC(IN.CAN(JOB),J)=0  GO LOOP
     ELSE ADD (DAM.REC(WO,J)-1.0) TO DAM.REC(WO.NUM(CAN.REC(IN.CAN(JOB),J)),J)
     SUBTRACT 1 FROM CAN.NUM(CAN.REC(IN.CAN(JOB),J))
       IF CAN.NUM(CAN.REC(IN.CAN(JOB),J)) NE 0  GO ZERO  ELSE
       IF CAN.REC(IN.CAN(JOB),J) IS NOT IN WT.QUEUE  GO ZERO  ELSE
       LET JOB.MOVE=CAN.REC(IN.CAN(JOB),J)
       SCHEDULE A MOVE.REAR GIVEN JOB.MOVE AND LEVEL IN (BETA.F(A(7,1),
         A(7,2),9)*(T.ACTION(7,3)-T.ACTION(7,1)))+T.ACTION(7,1)  HOURS
     'ZERO'
       LET CAN.REC(IN.CAN(JOB),J)=0
   'LOOP' LOOP
   RETURN
END
```

```
ROUTINE DET.ALLOC
   DEFINE PEOPLE AS A REAL VARIABLE
   DEFINE LEV.ATT AS A VARIABLE
   DEFINE P.DET, LAMBDA AND X AS REAL VARIABLES
   FOR EACH MAINT.UNIT IN SUP.BN WITH NAME(MAINT.UNIT)>0, DO
     LET LAMBDA=1./SQRT.F(VEH.COUNT(MAINT.UNIT))
     LET P.DET=EXP.F(-(LAMBDA*D.FLOT(MAINT.UNIT)**ALFA))
     LET PEOPLE=NM.FOLKS(MAINT.UNIT)+NF.FOLKS(MAINT.UNIT)
     IF PEOPLE LE 2. PRINT 1 LINE WITH NAME(MAINT.UNIT) THUS
MAINT UNIT = * ALREADY DESTROYED
     GO LOOP OTHERWISE
     LET X=RANDOM.F(7)
     IF X LE P.DET PRINT 1 LINE WITH NAME(MAINT.UNIT) AND PEOPLE THUS
MAINT UNIT = * DETECTED THIS BATTLE   *** PERSONNEL PRESENT
     LET LEV.ATT=MAINT.UNIT
     CALL ATTACK GIVEN LEV.ATT
     LET PEOPLE=PEOPLE-NM.FOLKS(MAINT.UNIT)-NF.FOLKS(MAINT.UNIT)
     PRINT 1 LINE WITH PEOPLE THUS
     *** PEOPLE KILLED IN THIS ATTACK
     ELSE PRINT 1 LINE WITH NAME(MAINT.UNIT), P.DET, D.FLOT AND VEH.COUNT THUS
MAINT UNIT = * NOT DETECTED    P.DET = *.****  D.FLOT = **.*** VEH = ***.
     PRINT 1 LINE WITH PEOPLE THUS
     *** PERSONNEL PRESENT AT UNIT
     ALWAYS 'LOOP' LOOP
   RETURN
END
```

```
ROUTINE ATTACK GIVEN LEV.ATT
   DEFINE LEV.ATT AND SPEC.JOB AS VARIABLES
   DEFINE I AND LEV.MOVE AS VARIABLES
   DEFINE MIS, N.AUTO, N.ARM AS VARIABLES
   DEFINE N, F, LOF, LOM AS REAL VARIABLES
   LET MAINT.UNIT=LEV.ATT
   IF PK.PERS GT RANDOM.F(7) SUBTRACT 1. FROM NM.FOLKS(MAINT.UNIT) ALWAYS
   FOR EACH CREW IN SHOP(MAINT.UNIT) WITH OCCUPATION(CREW) NE DEAD, DO
     FOR I=1 TO N.FOLKS(CREW),DO
       IF PK.PERS GT RANDOM.F(7) ''GUY KILLED''
         IF MISSION(CREW)=AUTO  SUBTRACT 1. FROM NM.FOLKS(MAINT.UNIT)
         ELSE SUBTRACT 1. FROM NF.FOLKS(MAINT.UNIT)
         ALWAYS SUBTRACT 1. FROM N.FOLKS(CREW)
       ALWAYS LOOP
   LOOP
   LET M=NM.FOLKS(MAINT.UNIT)/2.
   LET F=NF.FOLKS(MAINT.UNIT)/2.
   LET N.ARM=TRUNC.F(F)
   LET N.AUTO=TRUNC.F(M)
   IF FRAC.F(M)>0. LET LOM=1. ALWAYS
   IF FRAC.F(F)>0. LET LOF=1. ALWAYS
   FOR EACH CREW IN SHOP(MAINT.UNIT) WITH OCCUPATION(CREW)=BUSY, DO
     IF MISSION(CREW)=AUTO
       IF N.AUTO>0 SUBTRACT 1 FROM N.AUTO
         LET N.FOLKS(CREW)=2.
       ELSE LET OCCUPATION(CREW)=DEAD
        FOR EACH REPAIR IN EV.S(I.REPAIR) WITH A.CREW=CREW, DO
          CANCEL THE REPAIR LET JOB=SPEC.REP(REPAIR)
          REMOVE JOB FROM AUTOMOTIVE(MAINT.UNIT)
         FILE JOB IN WS.QUEUE(MAINT.UNIT)
          DESTROY THE REPAIR       LOOP
       ALWAYS
     ELSE IF N.ARM>0 SUBTRACT 1 FROM N.ARM
       LET N.FOLKS(CREW)=2.
     ELSE LET OCCUPATION(CREW)=DEAD
         FOR EACH REPAIR IN EV.S(I.REPAIR) WITH A.CREW=CREW, DO
           CANCEL THE REPAIR  LET JOB=SPEC.REP(REPAIR)
           REMOVE JOB FROM ARMAMENT(MAINT.UNIT)
          FILE JOB IN WS.QUEUE(MAINT.UNIT)
           DESTROY THE REPAIR  LOOP
   REGARDLESS ALWAYS LOOP
   FOR EACH CREW IN SHOP(MAINT.UNIT) WITH OCCUPATION(CREW)=IDLE, DO
     IF MISSION(CREW)=AUTO
       IF N.AUTO>0 SUBTRACT 1 FROM N.AUTO  LET N.FOLKS(CREW)=2.
       ELSE LET OCCUPATION(CREW)=DEAD
       ALWAYS
     ELSE IF N.ARM>0 SUBTRACT 1 FROM N.ARM  LET N.FOLKS(CREW)=2.
       ELSE LET OCCUPATION(CREW)=DEAD
       ALWAYS
   ALWAYS LOOP
```

```
IF LOF=1. FOR EACH CREW IN SHOP(MAINT.UNIT) WITH MISSION(CREW)=ARM
  AND OCCUPATION(CREW) NE DEAD, DO
  ADD 1. TO N.FOLKS(CREW) GO DOWN   LOOP
'DOWN' ALWAYS
IF LOM=1. FOR EACH CREW IN SHOP(MAINT.UNIT) WITH MISSION(CREW)=AUTO
  AND OCCUPATION(CREW) NE DEAD, DO
  ADD 1. TO N.FOLKS(CREW) GO OUTN   LOOP
'OUTN'  ALWAYS
FOR EACH REPAIR IN EV.S(I.REPAIR) WITH LEV.REP(REPAIR)=MAINT.UNIT AND
  LOOP.CH(SPEC.REP(REPAIR))=0, DO
  CANCEL THE REPAIR
  LET LOOP.CH(SPEC.REP(REPAIR))=1
  RESCHEDULE THE REPAIR AT TIME.A(REPAIR)+0.042
LOOP
FOR EACH JOB IN ARMAMENT(MAINT.UNIT) LET LOOP.CH(JOB)=0
FOR EACH JOB IN AUTOMOTIVE(MAINT.UNIT) LET LOOP.CH(JOB)=0
FOR EACH DIAGNOSIS IN EV.S(I.DIAGNOSIS) WITH LEV.DIAG(DIAGNOSIS)=MAINT.UNIT
  AND LOOP.CH(SPEC.DIAG(DIAGNOSIS))=0, DO
  CANCEL THE DIAGNOSIS
  LET LOOP.CH(SPEC.DIAG(DIAGNOSIS))=1
  RESCHEDULE THE DIAGNOSIS AT TIME.A(DIAGNOSIS)+0.042
LOOP
FOR EACH DIAGNOSIS IN EV.S(I.DIAGNOSIS) WITH LEV.DIAG(DIAGNOSIS)=MAINT.UNIT
  LET LOOP.CH(SPEC.DIAG(DIAGNOSIS))=0
LET MAINT.UNIT=LEV.ATT
LET LEV.MOVE=MAINT.UNIT
IF NM.FOLKS(MAINT.UNIT) LE 1.
  FOR EACH JOB IN WS.QUEUE(MAINT.UNIT) WITH MOB.DAM(JOB)>0., DO
    REMOVE JOB FROM WS.QUEUE(MAINT.UNIT)
    LET SPEC.JOB=JOB
    SCHEDULE A MOVE.REAR GIVEN SPEC.JOB AND LEV.MOVE IN (BETA.F(A(7,1),
      A(7,2),9)*(T.ACTION(7,3)-T.ACTION(7,1)))+T.ACTION(7,1) HOURS
  IF JOB IS NOT IN WT.QUEUE
    FILE JOB IN WT.QUEUE(MAINT.UNIT)
  ALWAYS
  FOR EACH REPAIR IN EV.S(I.REPAIR) WITH SPEC.REP=JOB, DO
    CANCEL THE REPAIR LET OCCUPATION(A.CREW)=IDLE
    IF JOB IS IN ARMAMENT REMOVE JOB FROM ARMAMENT ALWAYS
    LET CREW=A.CREW  DESTROY THE REPAIR     LOOP
  LOOP
  FOR EACH JOB IN WP.QUEUE(MAINT.UNIT) WITH MOB.DAM(JOB)>0., DO
    REMOVE JOB FROM WP.QUEUE(MAINT.UNIT)
    FOR EACH PARTS.COME IN EV.S(I.PARTS.COME) WITH SPEC.PART=JOB
      CANCEL THE PARTS.COME
    LET SPEC.JOB=JOB
    SCHEDULE A MOVE.REAR GIVEN SPEC.JOB AND LEV.MOVE IN (BETA.F(A(7,1),
      A(7,2),9)*(T.ACTION(7,3)-T.ACTION(7,1)))+T.ACTION(7,1) HOURS
    FILE JOB IN WT.QUEUE(MAINT.UNIT)
  LOOP
  FOR EACH JOB IN WI.QUEUE(MAINT.UNIT) WITH MOB.DAM(JOB)>0., DO
```

```
      REMOVE THE JOB FROM WI.QUEUE(MAINT.UNIT)
      LET SPEC.JOB=JOB
      SCHEDULE A MOVE.REAR GIVEN SPEC.JOB AND LEV.MOVE IN (BETA.F(A(7,1),
        A(7,2),9)*(T.ACTION(7,3)-T.ACTION(7,1)))+T.ACTION(7,1) HOURS
      FILE JOB IN WT.QUEUE(MAINT.UNIT)
    LOOP
    FOR EACH DIAGNOSIS IN EV.S(I.DIAGNOSIS) WITH LEV.DIAG=MAINT.UNIT, DO
      LET JOB=SPEC.DIAG(DIAGNOSIS)
      IF MOB.DAM(JOB)>0. CANCEL THE DIAGNOSIS  DESTROY THE DIAGNOSIS
      ADD 1 TO INSPECTOR(MAINT.UNIT)
      LET SPEC.JOB=JOB
      SCHEDULE A MOVE.REAR GIVEN SPEC.JOB AND LEV.MOVE IN (BETA.F(A(7,1),
        A(7,2),9)*(T.ACTION(7,3)-T.ACTION(7,1)))+T.ACTION(7,1) HOURS
      FILE JOB IN WT.QUEUE(MAINT.UNIT)
    ALWAYS LOOP
  ALWAYS
  IF NF.FOLKS(MAINT.UNIT) LE 1.
    FOR EACH JOB IN WS.QUEUE(MAINT.UNIT) WITH  FP.DAM(JOB)>0., DO
      REMOVE JOB FROM WS.QUEUE(MAINT.UNIT)
      LET SPEC.JOB=JOB
      SCHEDULE A MOVE.REAR GIVEN SPEC.JOB AND LEV.MOVE IN (BETA.F(A(7,1),
        A(7,2),9)*(T.ACTION(7,3)-T.ACTION(7,1)))+T.ACTION(7,1) HOURS
    IF JOB IS NOT IN WT.QUEUE
      FILE JOB IN WT.QUEUE(MAINT.UNIT)
    ALWAYS
    IF JOB IS IN AUTOMOTIVE REMOVE JOB FROM AUTOMOTIVE ALWAYS
    FOR EACH REPAIR IN EV.S(I.REPAIR) WITH SPEC.REP=JOB, DO
      CANCEL THE REPAIR  LET OCCUPATION(A.CREW)=IDLE
      DESTROY THE REPAIR LOOP
    LOOP
    FOR EACH JOB IN WP.QUEUE(MAINT.UNIT) WITH  FP.DAM(JOB)>0., DO
      REMOVE JOB FROM WP.QUEUE(MAINT.UNIT)
      FOR EACH PARTS.COME IN EV.S(I.PARTS.COME) WITH SPEC.PART=JOB
        CANCEL THE PARTS.COME
      LET SPEC.JOB=JOB
      SCHEDULE A MOVE.REAR GIVEN SPEC.JOB AND LEV.MOVE IN (BETA.F(A(7,1),
        A(7,2),9)*(T.ACTION(7,3)-T.ACTION(7,1)))+T.ACTION(7,1) HOURS
      FILE JOB IN WT.QUEUE(MAINT.UNIT)
    LOOP
    FOR EACH JOB IN WI.QUEUE(MAINT.UNIT) WITH  FP.DAM(JOB)>0., DO
      REMOVE THE JOB FROM WI.QUEUE(MAINT.UNIT)
      LET SPEC.JOB=JOB
      SCHEDULE A MOVE.REAR GIVEN SPEC.JOB AND LEV.MOVE IN (BETA.F(A(7,1),
        A(7,2),9)*(T.ACTION(7,3)-T.ACTION(7,1)))+T.ACTION(7,1) HOURS
      FILE JOB IN WT.QUEUE(MAINT.UNIT)
    LOOP
    FOR EACH DIAGNOSIS IN EV.S(I.DIAGNOSIS) WITH LEV.DIAG=MAINT.UNIT, DO
      LET JOB=SPEC.DIAG(DIAGNOSIS)
      IF  FP.DAM(JOB)>0. CANCEL THE DIAGNOSIS  DESTROY THE DIAGNOSIS
      ADD 1 TO INSPECTOR(MAINT.UNIT)
```

```
          LET SPEC.JOB=JOB
          SCHEDULE A MOVE.REAR GIVEN SPEC.JOB AND LEV.MOVE IN (BETA.F(A(7,1),
            A(7,2),9)*(T.ACTION(7,3)-T.ACTION(7,1))+T.ACTION(7,1) HOURS
          FILE JOB IN WT.QUEUE(MAINT.UNIT)
        ALWAYS LOOP
      ALWAYS
      RETURN
   END
```

```
ROUTINE TO COMP.TIMES
   DEFINE I AND J AS VARIABLES
   DEFINE B AND MU AS REAL VARIABLES
   PRINT 3 LINES AS FOLLOWS

INPUT TIME PARAMETERS

   FOR I=1 TO 8, DO
     FOR J=1 TO 3 READ T.ACTION(I,J)
     LET B=(T.ACTION(I,2)-T.ACTION(I,1))/(T.ACTION(I,3)-T.ACTION(I,1))
       LET MU=((4.0*B)+1.0)/6.0
       LET A(I,1)=((MU**2)*(1.0-MU)*36.0)-MU-1.0
     LET A(I,2)=((A(I,1)+1.0)/MU)-A(I,1)-1.0
   PRINT 1 LINE WITH I, A(I,1), A(I,2), T.ACTION(I,1), T.ACTION(I,2),
     T.ACTION(I,3)        THUS
   I=*  A(1)=****.**  A(2)=****.**  T.A(1)=**.**  T.A(2)=**.**  T.A(3)=**.**
   LOOP
   RETURN
END
```

```
ROUTINE INIT.PRINT
   PRINT 13 LINES WITH P.TANK, P.MOB, N.BNS, P.FIX.FWD, PR.HAVE.PARTS.FWD,
     P.CO.FIX, PR.REAR.HAVE.PARTS AS FOLLOWS

INPUT PARAMETERS

   THE PROPORTION OF JOBS THAT ARE TANKS IS .****
       THE REST ARE APC'S
   THE PROPORTION OF SYSTEM FAILURES THAT ARE AUTOMOTIVE IS .****
       THE REST ARE ARMAMENT JOBS
   THERE ARE * SUPPORTED BATTALIONS IN THE BRIGADE
   THE PERCENTAGE OF DAMAGE THAT CAN BE FIXED FORWARD IS .****
   THE PROBABILITY THAT THE FORWARD DET. WILL HAVE THE PARTS IS .****
   THE PERCENTAGE OF DAMAGE THAT CAN BE FIXED AT THE COMPANY IS .****
   THE PROBABILITY THAT THE COMPANY HAS THE PARTS IS .****

   PRINT 26 LINES WITH EX.RAT, BZERO, RZERO, BP, R.2ECH, SPACE.ECH,
     TGT.PRI, PR.INC.ID, LS, REC.NUM, SELF.LIKE, UNREC, TH, MCPD, CCSL, CCSU,
     MTTF, LEAD.TIME, SETUP.TIME, CON.SPEED, B.DIST, PK.PERS THUS

BATTLE AND RECOVERY INPUT PARAMETERS

   EXCHANGE RATIO AT BEGINNING OF BATTLE *.***
   BLUE FORCE LEVEL AT START OF BATTLE  ***.
   RED FORCE LEVEL AT START OF BATTLE  ***.
   RED BREAK POINT IS *.** SURVIVING
   RED SECOND ECHELON ADVANCE RATE **.* KM/HR
   ECHELON SPACING **. KM
   RECOVERY VEHICLE TARGET PRIORITY  *
   PROBABILITY OF INCORRECT IDENTIFICATION OF RECOVERY VEHICLE  .**
   PROBABILITY OF LINE OF SIGHT .**
   NUMBER OF RECOVERY VEHICLES AT START  **
   PROPORTION OF SELF AND LIKE RECOVERY  .**
   PROPORTION OF UNRECOVERABLE VEHICLES  .**
   EXPECTED HOOKUP TIME ***.* HOURS
   DISTANCE FROM BATTLE SITE TO MCP AT START OF BATTLE  ** KM
   CROSS COUNTRY SPEED LOADED **.* KM/HR
   CROSS COUNTRY SPEED UNLOADED **.* KM/HR
   MEAN TIME BETWEEN SYSTEM FAILURES  ***.* OPERATING HOURS
   WARNING TIME BEFORE START OF BATTLE *** HOURS
   TIME FOR SETUP AFTER MOVE  *** MINUTES
   CONVOY SPEED DURING MOVE **.** KM/HR
   BREAKPOINT DISTANCE AT WHICH DET. MOVES  ** KM
   PROBABILITY OF PERSONNEL CASUALTY IN ATTACK  *.**

   RETURN
END
```

```
ROUTINE GAMMA.F(MEAN,K,STREAM)
   DEFINE MEAN,K,KK,I,Z,A,B,D,E,X,Y AND W AS REAL VARIABLES
   DEFINE STREAM AS AN INTEGER VARIABLE
   IF MEAN LE 0., LET ERR.F=145 ELSE
   IF K LE 0.0  LET ERR.F=146 ELSE
   LET Z=0.
   REGARDLESS ALWAYS
   LET KK=TRUNC.F(K)
   LET D=K-KK
   IF KK=0.  GO TO BETA
   ELSE  LET E=1.0
   FOR I=1 TO KK  LET E=E*RANDOM.F(STREAM)
   LET Z=-LOG.E.F(E)
   IF D=0.  RETURN WITH Z*(MEAN/K)  ELSE
   'BETA'
   LET A=1.0/D  LET B=1.0/(1.0-D)
   'NEXT'
   LET X=RANDOM.F(STREAM)**A
   LET Y=RANDOM.F(STREAM)**B+X
   IF Y LE 1.0  GO OUT
   ELSE GO TO NEXT
   'OUT'
   LET W=X/Y
   LET Y=-LOG.E.F(RANDOM.F(STREAM))
   RETURN WITH (Z+W*Y)*(MEAN/K)
END
```

LIST OF REFERENCES

1. United States Army Field Manual 29-2, Organizational Maintenance Operations, Field Units, 29 August 1975.

2. United States Army Technical Manual 38-750, The Army Materiel Management System, 4 June 1978.

3. United States Army Field Manual 29-30-1, Divisional Maintenance Battalion, 20 February 1976.

4. United States Army Field Manual 29-25, Direct Exchange, Shop Supply and Operational Readiness Float Procedures, 3 March 1972.

5. United States Army Table of Organization and Equipment 29-2165700, Table of Organization and Equipment for the Brigade Support Battlion, DIV 86, November 1979.

6. United States Army Field Manual 100-5, Operations, 1 July 1976.

7. United States Army Logistics Center AURS 29-218S700, Operational Concept for the Division Support Command, 20 November 1979.

8. United States Army Concepts Analysis Agency Report CAA-TP-79-1, Methodology to Determine Support and Sustainability Implications of Increased POMCUS Levels, 30 June 1979.

9. United States Army Logistics Center Report FR-77-011, Evaluation of Maintenance Support Concepts, Braddock, Dunn and MacDonald, 18 February 1977.

10. Kiviat, P.J., Villanueva, R., Markowitz, H.M., SIMSCRIPT II.5 Programming Language, C.A.C.I., 1975.

11. Affeldt, A.F., Parametric Analysis of Recovery and Evacuation of Tracked Vehicles, Masters Thesis, Naval Postgraduate School, Monterey, California, 1980.

12. United States Army Logistics Center, Maintenance Concepts Simulation of Restructured General Support Final Report, by the Operations Analysis Directorate, U.S. Army Logistics Center, July 1978.

13. Hagewood, E.G. and Wallace, W.S., Simulation of Tactical Alternative Responses (STAR), Masters Thesis, Naval Postgraduate School, Monterey, California, December 1978.

14. Taylor, J.G. and Comstock, C., "Force-Annihilation Conditions for Variable-Coefficient Lanchester-Type Equations of Modern Warfare", Naval Research Logistics Quarterly, v.24, p. 349-371, 1977.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
3.	Department Chairman, Code 55 Department of Operations Research Naval Postgraduate School Monterey, California 93940	1
4.	Chief TRADOC Research Element Monterey Naval Postgraduate School Monterey, California 93940	1
5.	CPT Andrew G. Loerch 34 Meagher Place Williston Park, New York 11596	1
6.	Professor James K. Hartman, Code 55Hh Department of Operations Research Naval Postgraduate School Monterey, California 93940	10
7.	Professor S. H. Parry, Code 55Py Department of Operations Research Naval Postgraduate School Monterey, California 93940	10
8.	MAJ Jeffrey Ellis, Code 55Ei Department of Operations Research Naval Postgraduate School Monterey, California 93940	1
9.	Professor Arthur L. Schoenstadt, Code 53Zh Department of Mathematics Naval Postgraduate School Monterey, California 93940	1

10. Office of the Commanding General | 1
U. S. Army TRADOC
ATTN: General Donn A. Starry
Ft. Monroe, Virginia 23651

11. Headquarters | 1
U. S. Army Training & Doctrine Command
ATTN: Director, Analysis Directorate
(COL D. Maddox)
Ft. Monroe, Virginia 23651

12. Headquarters | 1
U. S. Army Training & Doctrine Command
ATTN: Director, Analysis Directorate
Combat Developments (MAJ Chris Needels)
Ft. Monroe, Virginia 23651

13. Mr. David Hardison | 1
Deputy Under Secretary of the Army
(Operations Research)
Department of the Army, The Pentagon
Washington, D. C. 20310

14. LTG William Richardson | 1
Commanding General
U. S. Army Combined Arms Center
Ft. Leavenworth, Kansas 66027

15. Director | 1
Combined Arms Combat Development Activity
ATTN: Col. Reed
Ft. Leavenworth, Kansas 66027

16. Director | 1
Combat Analysis Office
ATTN: Mr. Kent Pickett
U. S. Army Combined Arms Center
Ft. Leavenworth, Kansas 66027

17. Command and General Staff College | 1
ATTN: Education Advisor
Room 123, Bell Hall
Ft. Leavenworth, Kansas 66027

18. Dr. Wilbur Payne, Director | 1
U. S. Army TRADOC Systems Analysis Activity
White Sands Missile Range, New Mexico 88002

19. Headquarters, Department of the Army 1
Office of the Deputy Chief of Staff
for Operations and Plans
ATTN: DAMO-2D
Washington, D. C. 20310

20. Commander 1
U. S. Army Concepts Analysis Agency
8120 Woodmont Avenue
ATTN: MOCA-SMS (CPT Steve Shupack)
Bethesda, Maryland 20014

21. Commander 1
U. S. Army Concepts Analysis Agency
ATTN: MOCA-WG (LTC Earl Darden)
8120 Woodmont Avenue
Bethesda, Maryland 20014

22. Director 1
U. S. Army Night Vision & Electro-optical Lab.
ATTN: DEL-NV-VI-Mr. Bob Hermes
Ft. Belvoir, Virginia 22060

23. Director 1
U. S. Army Material Systems Analysis Activity
ATTN: Mr. Will Brooks
Aberdeen Proving Grounds, Maryland 21005

24. Col. Frank Day 1
TRADOC Systems Manager - XM1
U. S. Army Armor Center
Ft. Knox, Kentucky 40121

25. Director 1
Combat Developments, Studies Division
ATTN: MAJ W. Scott Wallace
U. S. Army Armor Agency
Ft. Knox, Kentucky 40121

26. Commandant 1
U. S. Army Field Artillery School
ATTN: ATSF-MBT (CPT Steve Starner)
Ft. Sill, Oklahoma 73503

27. Director, Combat Developments 1
ATTN: Col. Clark Burnett
U. S. Army Aviation Agency
Ft. Rucker, Alabama 36362

28. Director 1
Combat Developments
U. S. Army Infantry School
Ft. Benning, Georgia 31905

29. Commander, MICOM 1
ATTN: DRSMI-YC (MAJ Hagewood)
Redstone Arsenal, Alabama 35809

30. Director 1
Combat Developments
ATTN: MAJ William D. Meiers
U. S. Army Air Defense Agency
Ft. Bliss, Texas 79905

31. Commander 1
U. S. Army Logistics Center
ATTN: ATCL-OS-Mr. Cammeron/CPT Schuessler
Ft. Lee, Virginia 23801

32. Director 1
U. S. Army AMSAA
ATTN: DRXSY-AA (Mr. Tom Coyle)
Aberdeen Proving Grounds, Maryland 21005

33. Office of the Deputy Chief of Staff 1
for Combat Developments
U.S. Army TRADOC
ATTN: Major General Carl Vuono
Fort Monroe, Virginia 23651

34. Deputy Commanding General 1
Combined Arms Combat Development Activity
ATTN: ATZL-CA-DC (MG Jack Walker)
Fort Leavenworth, Kansas 66027

35. Director 1
USATRASANA
ATTN: Mr. Ray Heath
White Sands Missile Range, New Mexico 88002

36. Commander 1
U.S. Army Logistics Center
ATTN: ATCL-CTD (LTC Arnold)
Fort Lee, Virginia 23801

DATE
FILMED

-8